Discovering Inconsistencies between Requested Permissions and Application Metadata by using Deep Learning

1st Huseyin Alecakir
Cognitive Science
Middle East Technical University
Ankara, Turkey
huseyinalecakir@gmail.com

2rd Muhammet Kabukcu
WISE Lab.
Hacettepe University
Ankara, Turkey
mkabukcu@gmail.com

3nd Burcu Can
RIILP
University of Wolverhampton
Wolverhampton, UK
b.can@wlv.ac.uk

4rd Sevil Sen
WISE Lab.
Hacettepe University
Ankara, Turkey
ssen@cs.hacettepe.edu.tr

***Abstract*—Android gives us opportunity to extract meaningful information from metadata. From the security point of view, the missing important information in metadata of an application could be a sign of suspicious application, which could be directed for extensive analysis. Especially the usage of dangerous permissions is expected to be explained in app descriptions. The permission-to-description fidelity problem in the literature aims to discover such inconsistencies between the usage of permissions and descriptions. This study proposes a new method based on natural language processing and recurrent neural networks. The effect of user reviews on finding such inconsistencies is also investigated in addition to application descriptions. The experimental results show that high precision is obtained by the proposed solution, and the proposed method could be used for triage of Android applications.**

***Index Terms*—Android, security, description-to-permission fidelity, deep learning, natural language processing, recurrent neural networks**

I. Introduction

Mobile devices have become inevitable part of our lives. With the use of applications, they provide many functionalities such as writing/reading emails, mobile banking, finding nearby facilities that make our lives easier. While Android is the leading mobile operating system with approximately 75% share in the market [1], it is also one of the most targeted platforms by attackers [2].

There are two main techniques in order to detect malware: static and dynamic analysis. While applications are run on a device or an emulator to observe the runtime behaviours of programs in dynamic analysis, in static analysis the application is not run and its code and other files in the application package such as manifest file are analyzed statically. Both techniques have their own pros and cons and both could be bypassed by evasive techniques. Besides code and malicious behaviours of applications, Android platform give us another data to analyze applications from the security point of view: metadata in the market stores such as descriptions, user reviews and ratings, privacy policies. Therefore, in the last five years, we have started to see the applications of natural language processing (NLP) in order to extract meaningful, security-related data from metadata of Android applications.

In this study, a new approach based on NLP and recurrent neural networks (RNNs) is employed in order to find inconsistencies between the requested permissions of an application and its metadata. This problem is known as description-to-fidelity problem [3] in the literature. Permissions [4] is one of the important security mechanisms introduced by Android, so users have to grant dangerous permissions before their usage. While dangerous permissions need to be granted at the time of installation before Android 6.0, they are asked to be granted at runtime in recent versions of Android.

We expect Android developers to give more information about the usage of dangerous permissions in their metadata such as descriptions and privacy policies. The missing of such an important data could be a sign of a suspicious activity, which could be forwarded for further static and dynamic analysis. This is the main assumption of the current study and the previous studies proposed for the description-to-fidelity problem. In this research, recurrent neural networks are used to detect whether a particular application needs permission. To this end, we suggest a model for the description text that utilizes long-term short-term memory (LSTM) networks. Two main contributions of the study can be summarized as follows. Firstly, a neural model using deep recurrent neural networks is proposed to detect the inconsistencies between requested permissions and descriptions. Secondly, the effects of reviews are explored for the description-to-permission fidelity problem.

The remainder of the paper is organized as follows: Section II summarizes the previous NLP-based studies in Android security in the literature. Section III introduces the method based on LSTMs proposed for the problem. The experimental results of the proposed approach on the AC-Net dataset [5] are given and discussed in Section IV. The effects of user reviews on the results are also discussed in this section. Finally, Section V concludes the paper.

II. Related Work

Google Play, Android official market, and other market stores provide a platform for users to search and download for applications. These platforms is also used by attackers

to distribute their malicious applications. In market stores, besides the application package, other information about applications such as descriptions, ratings, user reviews exist. Such information which is mainly based on text is called metadata. Here, we summarize the studies that use metadata for Android security, especially proposed for the permission-to-description fidelity problem.

WHYPER [6] is one of the earliest attempts made to automatically analyze description-to-permission fidelity of applications. WHYPER applies natural language processing methods to identify app descriptions that involve a specific permission and thereby describe the need for a permission. For this purpose, first-order logic representations of description sentences are built to match them with the semantic graphs of permissions. Another study by Watanabe et al. [7] proposes a keyword-based method for description-to-permission fidelity and achieves comparable results with WHYPER [6]. One of the main advantage of that study is that it can be easily applied to other languages.

Another model, AutoCog proposed by Qu et al. [3] is based on explicit semantic analysis (ESA) which provides a vectoral representation of a given text, thereby giving a representation of the meaning in the text. The model uses vectoral representations of app descriptions to assess their semantic relatedness to permissions. The results obtained from AutoCog are comparably better than that of WHYPER. A recent study by Feng et al. [5] is the closest one to our current study. AC-Net [5] is the only research that applied artificial neural networks before us in the problem of permission to description fidelity. In that study, AC-Net makes use of sequential-based neural networks, specifically gated recurrent unit(GRU) [8]. GRU [8] network is a type of recurrent neural works (RNNs) which are used to model sequential data that relate to each other. In the current study that is developed independently from AC-NET, a method based on LSTM is employed. Furthermore, differently from AC-NET, the effects of user reviews are explored.

To date, several studies have examined the impact of user reviews on Android app security and privacy features [9]–[11]. Kong et al. [9] proposed a system called AUTOREB to learn the privacy-related behaviors inferred from user reviews analysis. AUTOREB is a machine learning-based algorithm to associate the relations between reviews and privacy-related behaviors. A recent study by Nguyen et al. [10] presented a natural language-based model to classify the user reviews into security&privacy-related (SPR) or non-SPR. They showed that there is a strong correlation between SPR reviews and privacy-related app updates. A more recent study, [11], made use of user comments to understand why an application is asking for permissions. The proposed NLP based SmartPI [11] framework detects user comments related to functionality and extracts permissions associated with them. In SmartPI, the user reviews are not labelled.

Wang et al. [12] proposes a data mining-based model for understanding why applications ask for their permissions. They make use of the static analysis to extract permission related code features for their machine learning based classifiers. Finally, the classifiers outputs a categorical purpose of a given permission. There are also few studies that focused on automatic security-centric description generation [12], [13]. NLP has started to be used in malware detection as well. Yu et al. [14] proposed a malware detection system based on the privacy policy and static code analysis. Another study is conducted by Mu et al. [13] by leveraging the Natural Language Generation techniques to generate security-centric application descriptions bases on the program analysis.

TABLE I
INFORMATION ABOUT THE SELECTED PERMISSION GROUPS.

Protection Levels	Permission Groups	Permission
Dangerorus	STORAGE	WRITE_EXTERNAL_STORAGE
		GET_ACCOUNTS
	CONTACTS	READ_CONTACTS
		WRITE_CONTACTS
	LOCATION	ACCESS_FINE_LOCATION
		ACCESS_COARSE_LOCATION
	CAMERA	CAMERA
	MICROPHONE	RECORD_AUDIO
	SMS	READ_SMS
		SEND_SMS
	CALL_LOG	READ_CALL_LOGS
	PHONE	CALL_PHONE
	CALENDAR	READ_CALENDAR
Signature	SETTINGS	WRITE_SETTINGS
Normal	TASKS	GET_TASKS
		KILL_BACKGROUND_PROCESS

III. MODEL

The overall architecture of the proposed approach is given in Figure 1. In training, which is given on the right side of Figure 1, labelled application descriptions (i.e. sentences) are pre-processed, then feature vectors are built based on pre-trained word embeddings, and fed into a Long Short Term Memory Network (LSTM). An LSTM-based prediction model is built for each permission type. In testing, which is given on the left side of Figure 1, a new description is first parsed into sentences. Each sentence is pre-processed and its feature vector is built using pre-trained word embeddings. Then, each feature vector is fed into the trained prediction model for each permission. If the prediction score for a permission is larger than 0.5, the sentence is identified as a corresponding permission sentence.

For labelled descriptions of application, AC-Net [5] dataset is employed in this study. In this dataset, application sentences obtained from 1417 applications' descriptions are manually marked as for 11 permission groups. If the selected sentence is related to the relevant permission group, it is labeled as 1; if not, it is labeled as 0. Unlike previous studies in the literature [3], [6], AC-Net labelled application descriptions according to permission groups rather than single permission. 9 of these permission groups are belong to dangerous permissions. Besides the dangerous permissions, 2 permission groups are belong to the signature and normal permissions. Table I shows the permission groups used in the AC-Net study, the

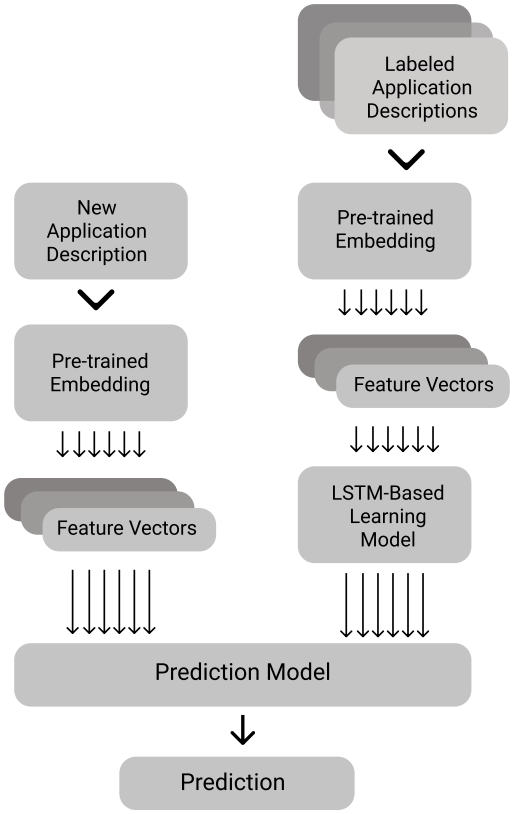

Fig. 1. Model Overview

TABLE II
AC-NET DATASET STATISTICS. S : SENTENCES PS : PERMISISON SENTENCES.

Permission Groups	# of Apps	# of S	# of PS
STORAGE	1304	23101	1338
CONTACTS	951	17353	937
LOCATION	732	12887	724
CAMERA	406	7372	522
MICROPHONE	350	6371	319
SMS	337	6484	524
CALL_LOG	282	5457	323
PHONE	280	5445	199
CALENDAR	197	3637	289
SETTINGS	369	7016	560
TASKS	538	10203	344
TOTAL	1415	24726	4984

permissions that these permission groups correspond and the number of applications in each group. Here, also the number of sentences and, the number of permission sentences that are marked as 1 are also shown.

Unlike Feed-Forward Neural Networks, Recurrent Neural Networks (RNNs) [15] have the ability to process sequential data with an internal memory that allows to remember history while processing future inputs. Due to the vanishing gradient problem in RNNs, Long Short Term Memory Networks (LSTMs) [16] filter out relevant information while remembering past information which also provides such networks to overcome vanishing gradients problem. We utilize LSTMs in our approach for processing app descriptions that are composed of sentences and sequential by nature.

First, app descriptions are pre-processed before feeding into the LSTMs. The pre-processing tasks involve sentence tokenization, word tokenization, punctuation removal, stopwords elimination, non-alpha characters removal, and stemming[1]. Then embeddings of words are obtained from pre-trained word embeddings and these embeddings of each sentence are fed into an LSTM. The output of each LSTM gives a compositional representation of the input sentence that involves its semantics. We use a bidirectional LSTM, where one LSTM processes the words in a sentence in forward order, and another LSTM processes the words in a sentence in reverse order:

$$s_i = BiLSTM(x_{1:n}, i) \tag{1}$$

Here, $s_i = \{x_1, \cdots, x_n\}$ is the ith sentence in the data and x_j is the embedding of the jth word in the sentence. We use s_i for the compositional representation of the sentence. The final hidden vectors of both LSTMs are concatenated to have the sentence representation s_i. Once the compositional representation of each sentence is obtained from the bidirectional LSTM, each sentence is classified using a multilayer perceptron with a sigmoid activation function as given below:

$$o_i = \text{sigmoid}(MLP(s_i)) \tag{2}$$

where o_i refers to the prediction output, which is called the permission score here. It is a value between 0 and 1 in order to decide whether the corresponding permission is stated in the given input. If it is bigger than 0.5, the classification output is 1, which indicates that the permission is explained in the sentence. Otherwise, the output is 0, hence the usage of the permission is not stated in the sentence.

Here, the problem is a binary classification problem. Hence, a different LSTM model is produced for each permission separately. In training, binary cross-entropy is used as a loss function in order to measure the loss between the target output and the predicted output in each LSTM:

$$= -(y * \log(\hat{y}) + (1 - y) * \log(1 - \hat{y})) \tag{3}$$

Here, y refers to the target output and $\hat{y}$ refers to the predicted output. The architecture of the model is given in Figure 2.

The model is implemented in DyNet library[2][3]. The dimension of word embeddings is 300. Therefore, each LSTM has an input dimension of 300 and hidden layer dimension of 128. The MLPs have a hidden layer dimension of 128 and output dimension of 1 where an output of 1 indicates that the permission is stated in the sentence, and an output of 0 indicates that the permission is not mentioned in the given description sentence. We randomly initialize the model parameters from a uniform distribution in the range of 0.08 and

[1]We use Porter stemmer [17].
[2]https://dynet.readthedocs.io/en/latest/tutorial.html
[3]The implementation will be publicly available if the paper gets accepted.

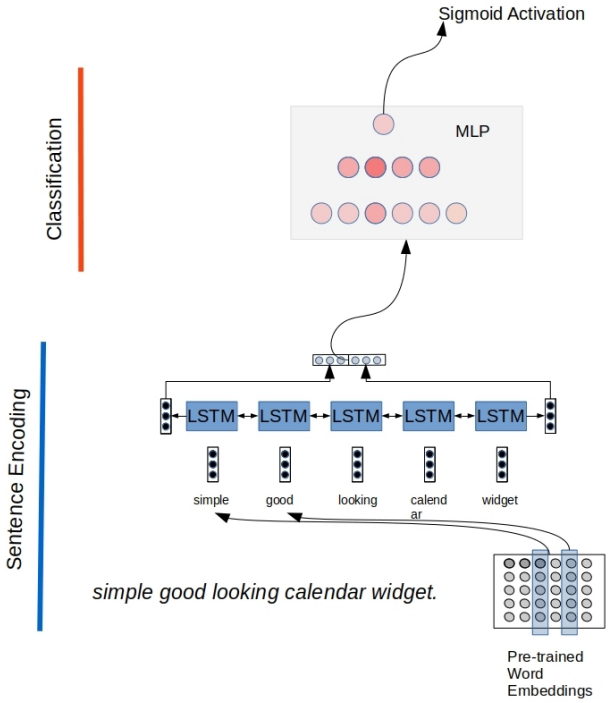

Fig. 2. BI-LSTM based classification.

TABLE III
EVALUATION SCORES OF THE PROPOSED MODEL ON THE AC-NET DATASET

Permission Group	Fasttext		Domain adapted	
	ROC-AUC	PR-AUC	ROC-AUC	PR-AUC
CONTACTS	0.96	0.71	0.97	0.73
MICROPHONE	0.88	0.33	0.94	0.39
CALENDAR	0.97	0.73	0.98	0.76

0.08. We use Adaptive Moment Estimation (Adam) optimizer [18]. We apply gradient norm clipping to deal with exploding gradient problem [19]. We use 10-fold cross-validation for training.

IV. EXPERIMENTS

A. Evaluation Metrics

In this study, we used the PR-AUC and ROC-AUC metrics as the evaluation metrics. Since the AC-Net dataset [5] dataset is not well balanced, we used the PR-AUC and ROC-AUC metrics instead of standard metrics such as accuracy. To give an example from the camera permission, only 522 of the 24724 sentences in the AC-NET dataset seem to be related to the CAMERA permission.

B. Results without Using Reviews

In this study, we initialize word vectors with our pre-trained embeddings. We apply the skip-gram word2vec algorithm [20], [21] in order to produce our domain adapted word embedding from collected descriptions of Android applications. In loosely speaking, word vectors correspond to the meaning of words, and they could have different meanings in different domains. Here, we have created the word vectors from the Android context to in order to eliminate the ambiguity that results from other meaning of words in other domains. The results of the proposed approach are compared with Fasttext embeddings, which is a type of unsupervised learning algorithm for obtaining vector representations for words, in Table III. It is clearly seen that the accuracy increases by using the domain adapted word embeddings. The results we have obtained by integrating our own embedding are shown in Table IV. The AC-Net study similarly produced its own word embedding. As can be seen in Table IV, we got comparable results with the AC-Net study.

TABLE IV
COMPARISON WITH AC-NET

Permission Group	AC-NET		Our Model	
	ROC-AUC	PR-AUC	ROC-AUC	PR-AUC
CONTACTS	0.97	0.75	0.97	0.72
MICROPHONE	0.96	0.50	0.96	0.43
CALENDAR	0.99	0.84	0.99	0.80
ACCESS_FINE_ LOCATION	0.98	0.77	0.98	0.75
CALL_PHONE	0.99	0.63	0.99	0.57
CAMERA	0.98	0.76	0.98	0.70
GET_TASKS	0.95	0.48	0.93	0.47
READ_CALL_LOGS	0.99	0.71	0.99	0.57
READ_SMS	0.99	0.84	0.99	0.80
WRITE_SETTINGS	0.95	0.43	0.95	0.44

We analyzed the sample of erroneous cases to determine the capacity and effectiveness of our model. Here, we compare the proposed approach with AC-Net dataset. We did this analysis only on the CONTACTS permission. Of the 24726 sentences in the AC-Net dataset, only 937 have been shown to be associated with the CONTACTS permission. Our model that we trained correctly classifies 99.01 out of 24726 sentences.

In this section, we will explain the reasons for the erroneous classifications. Our model can distinguish the meaning of the same word in different contexts. For instance, our model differentiates the meaning of ”contact” keyword in a permission sentence (e.g. ”share your contact with your friends”) and in a statement sentence (e.g. ”Please contact us at team loves to hear from its users”). However, our model does not have adequate complexity in order to classify the following sentence ”This app helps you to find apps that requests read contacts permission”. In this example, even the candidate app does not request the mentioned permission, our model classifies it as a permission sentence. We require more complex models that involve natural language understanding for such sentences.

Finally, some inconsistencies arising from the data caused the model to produce erroneous results. This situation is very common in cases containing information about social media. For example, phrases such as ”share with your friends via Facebook” are sometimes marked as permission clauses and sometimes as normal sentences. In training and prediction step, such labelling errors might lead to contradictory results, thereby increasing both false positives and false negatives.

TABLE V
EFFECTS OF USER REVIEWS

Permission Group	Without Reviews		With Reviews	
	ROC-AUC	PR-AUC	ROC-AUC	PR-AUC
READ_CONTACTS	0.57	0.41	0.69	0.53
RECORD_AUDIO	0.48	0.15	0.61	0.21
STORAGE	0.65	0.56	0.51	0.47

C. Results of the Document-based Model Using Reviews

One of the main reasons for developing the description-based model in this study is to evaluate the contribution of user reviews for solving the permission-to-description fidelity problem. In order to evaluate the effects of reviews, a document-based model is trained and tested on the AC-Net dataset. Using the same LSTM model which is trained on the description dataset, we employed the reviews in testing. However, this time we used the full description while feeding into the LSTM, rather than feeding the sentences within the description one by one. Therefore, we train the model using the descriptions each as a single document.

In this study, user reviews are also treated as a text like a description. In our document based model, we followed the following strategy. Once the model is trained, we test each description in the test set whether a permission is stated or not. If the permission score is high and the permission is detected in the description, then it is concluded that the permission is required by the application. However, if the permission score is low, then it is still possible that the permission is required by the application but it could be ignored in the description. This time we feed the most useful 3 user reviews of the same application into the trained LSTM network and we concatenate encoding of the description LSTM with encoding of the review LSTM to predict the permission score. The permission scores are expected to change with the inclusion of the reviews in testing, if the permission is indeed required by the application. The results once the reviews are included in testing are given in Table V. Here, the threshold of the permission score is taken as 0.5. The results show the positive effects of reviews, especially for the READ_CONTACTS and RECORD_AUDIO permissions. However, as it is noted, the performance of the document-based is considerably lower than the sentence-based model. The main reason of that is the number of training samples. Because the number of documents, i.e. applications, is considerably smaller than the total number of sentences in the dataset. This effect is also observed for the STORAGE permission. In the future, we aim to increase the size of the dataset for evaluating a document-based model.

V. CONCLUSION AND FUTURE WORK

Description-to-fidelity problem refers to the inconsistencies between requested permissions and application metadata. In this paper, we use natural language processing methods, as well as recurrent neural networks to tackle the description-to-fidelity problem in Android applications. Our results show that using a basic bidirectional LSTM network detects the inconsistencies between the requested permissions and application descriptions reasonably.

Our model is similar to the recent neural model AC-Net [5] since both use recurrent neural networks. However, their model is based on Gated Recurrent Units (GRU), whereas ours uses LSTMs. Another difference between their model and ours is the inclusion of the user reviews. It is possible to have some descriptions that do not mention about the permissions, but still it is possible to catch those descriptions through the user reviews. Our results show that using reviews could improve the scores.

We plan to further investigate the methods to include the reviews in the sentence-based model, which is left as a future goal.

Acknowledgements This study is supported by the Scientific and Technological Research Council of Turkey (TUBITAK-118E141). The authors would like to thank TUBITAK for its support.

REFERENCES

[1] Statista. (2020) Mobile operating systems' market share worldwide from january 2012 to july 2020. (Visited November 2020) [Online]. Available: https://www.statista.com/statistics/272698/global-market-share-held-by-mobile-operating-systems-since-2009/.

[2] Symantec. (February 2019) Internet security threat report. https://www.symantec.com/.

[3] Z. Qu, V. Rastogi, X. Zhang, Y. Chen, T. Zhu, and Z. Chen, "Autocog: Measuring the description-to-permission fidelity in android applications," in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '14. New York, NY, USA: ACM, 2014, pp. 1354–1365. [Online]. Available: http://doi.acm.org/10.1145/2660267.2660287

[4] "Permissions overview," https://developer.android.com/guide/topics/permissions/overvie 2019, (Last accessed in May, 2019).

[5] Y. Feng, L. Chen, A. Zheng, C. Gao, and Z. Zheng, "Ac-net: Assessing the consistency of description and permission in android apps," *IEEE Access*, vol. 7, pp. 57 829–57 842, 2019.

[6] R. Pandita, X. Xiao, W. Yang, W. Enck, and T. Xie, "Whyper: Towards automating risk assessment of mobile applications," in *Proceedings of the 22Nd USENIX Conference on Security*, ser. SEC'13. Berkeley, CA, USA: USENIX Association, 2013, pp. 527–542. [Online]. Available: http://dl.acm.org/citation.cfm?id=2534766.2534812

[7] T. Watanabe, M. Akiyama, T. Sakai, and T. Mori, "Understanding the inconsistencies between text descriptions and the use of privacy-sensitive resources of mobile apps," in *Eleventh Symposium On Usable Privacy and Security (SOUPS 2015)*. Ottawa: USENIX Association, 2015, pp. 241–255. [Online]. Available: https://www.usenix.org/conference/soups2015/proceedings/presentation/watanabe

[8] K. Cho, B. van Merrienboer, Ç. Gülçehre, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," *CoRR*, vol. abs/1406.1078, 2014. [Online]. Available: http://arxiv.org/abs/1406.1078

[9] D. Kong, L. Cen, and H. Jin, "Autoreb: Automatically understanding the review-to-behavior fidelity in android applications," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2015, pp. 530–541.

[10] D. C. Nguyen, E. Derr, M. Backes, and S. Bugiel, "Short text, large effect: Measuring the impact of user reviews on android app security privacy," 03 2019.

[11] R. Wang, Z. Wang, B. Tang, L. Zhao, and L. Wang, "Smartpi: Understanding permission implications of android apps from user reviews," *IEEE Transactions on Mobile Computing*, 2019.

[12] H. Wang, Y. Li, Y. Guo, Y. Agarwal, and J. I. Hong, "Understanding the purpose of permission use in mobile apps," *ACM Transactions on Information Systems (TOIS)*, vol. 35, no. 4, p. 43, 2017.

[13] M. Zhang, Y. Duan, Q. Feng, and H. Yin, "Towards automatic generation of security-centric descriptions for android apps," in *Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '15. New York, NY, USA: ACM, 2015, pp. 518–529. [Online]. Available: http://doi.acm.org/10.1145/2810103.2813669

[14] L. Yu, X. Luo, C. Qian, S. Wang, and H. K. Leung, "Enhancing the description-to-behavior fidelity in android apps with privacy policy," *IEEE Transactions on Software Engineering*, vol. 44, no. 9, pp. 834–854, 2017.

[15] J. Elman, "Finding structure in time," *Cognitive science*, vol. 14, no. 2, pp. 179–211, 1990.

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

[17] M. F. Porter, "Readings in information retrieval," K. Sparck Jones and P. Willett, Eds. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1997, ch. An Algorithm for Suffix Stripping, pp. 313–316. [Online]. Available: http://dl.acm.org/citation.cfm?id=275537.275705

[18] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014.

[19] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," in *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28*, ser. ICML'13. JMLR.org, 2013, pp. III–1310–III–1318. [Online]. Available: http://dl.acm.org/citation.cfm?id=3042817.3043083

[20] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in Neural Information Processing Systems 26*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2013, pp. 3111–3119.

[21] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.