International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-9 Issue-2, December, 2019

Deep Learning and NLP based Side Channel Attack for Text Inference in Smartphones

P Uma Maheswari, Mohamed Yilmaz Ibrahim, Ramkumar B, Aswin Sundar

***Abstract**: Over the past years, smartphones have witnessed an alarming rise in embedded sensors which enhance their support for applications. However, they can be regarded as loopholes as seemingly innocuous information can be obtained without any user permissions in Android thus invading the user's privacy. Our work establishes a side channel attack by illegitimately inferring the information being typed by the user on a smartphone using the readings from 'zero-permission' sensors like accelerometer and gyroscope. This serves as a proof of concept to prevent such attacks on mobile devices in the future. While previous research has been conducted in this space, our narrative involves a predictive model using Recurrent Neural Networks that can predict the letters being typed in the keyboard solely based on the motion sensor readings, thus inferring the text. Our research was able to identify 37.5% of the unseen words typed by the user using a very small volume of training data. Our tap detection method has shown 92% accuracy which plays a critical role in the text inference. This research lays the foundation to further progress in this area, thus helping to strengthen the mobile security.*

Index Terms— Android, Security, Side-channel attack, LSTM

I. INTRODUCTION

The usage of smart mobile devices for personal and business purposes has seen immense rise in popularity over the last decade. From communication to payments, mobile devices have applications in almost all domains. This drastic shift in the usage of mobile devices has increased the amount of potentially sensitive material and activity performed on them. These smartphones have become increasingly personal and thus privacy has become a crucial issue and much research has been performed on the permissions model governing them. Our work explores one particular way to bypass this security model such that one application can read the data being typed in another application.

Sensors like gyroscope, accelerometer and orientation sensors have originally been designed to monitor a user's location, movement, orientation, altitude and other such potential information. However, previous research has confirmed that motion sensors can act as a side channel for inferring the user's keystroke or input information on smartphones. Thus, applications are specifically being designed by attackers to collect data from these motion sensors and perform text inference attacks with the help of machine learning algorithms. This can prove critical to the users as even their sensitive information such as passwords or credit card information can be extracted.

The main objectives of this work are as follows:

1) To prove that an app in the background can infer the information being typed in another application through the sensor readings.

2) To employ Deep Learning and Natural Language Processing techniques to deduce the typed information.

We employ traditional classification methods such as RMSE as well as deep neural networks to infer the typed sentences. By grouping keyboard keys into larger regions, the tap position can be determined more accurately and a language model is used to localize the region into one of the keys thus improving the overall inference.

This paper continues in Section 2 with a discussion of the academic background to this research, Section 3 then explores the system architecture and the design whilst Section 4 discusses the analysis of the experimental data. Finally, the conclusion of the paper is explored in Section 5.

II. RELATED WORK

According to the work by Genkin et al. [1], the scope of applications in smartphones has seen a drastic increase and as a result they have become more personal making us inseparable from our smartphones. Thus, it becomes crucial for us to secure the mobile devices. TouchLogger [2] was a smartphone application designed to serve the purpose of inferring the keystrokes made on a soft keyboard based exclusively on the vibrations recorded by the smartphone's motion sensors. Their research had successfully inferred more than 70% of the keystrokes using only the accelerometer sensor of the device. However, this work had a restriction as it has been focused specifically on inferring the keystrokes from a numeric keyboard. Similarly, Xu et al. present TapLogger [3], an approach that looks to infer an individual's taps on a numeric keyboard using a smartphone's accelerometer and gyroscope. This work has enhanced functionality as it had laid attention on identifying single taps, which are more susceptible to distortion by linear drift.

This paved a way for more active research in this area to not only infer the numbers detected from the keyboard but also text. The work of Aviv et al. [4] builds on the idea of PIN identification using motion sensors.

Revised Manuscript Received on December 30, 2019.
* Correspondence Author
Dr. P Uma Maheswari, School of Computer Science and Engineering, CEG, Anna University, Chennai, India, Email: dr.umasundar@gmail.com
Mohamed Yilmaz Ibrahim*, School of Computer Science and Engineering, CEG, Anna University, Chennai, India.
Email: mohamedyilmaz98@gmail.com
Ramkumar B, School of Computer Science and Engineering, CEG, Anna University, Chennai, India. Email: therealramkumar@gmail.com
Aswin Sundar, School of Computer Science and Engineering, CEG, Anna University, Chennai, India. Email: aswinsundar17@gmail.com

Contrary to the single tap identification established by previous works, their approach brings out the notion of detecting and inferring the swipe gestures of a victim's PIN. The other approach has focused on password compromise, for example, Owusu et al. [5] have used a smartphone's accelerometer to infer the characters, both letters and numbers, contained within a user's password. Our approach looks to identify the text which has been inputted by the user on a qwerty keyboard which provides a much more robust identification method. The work of Miluzzo et al [6] on Tap Prints builds on this idea of keypress identification by attempting to determine the tap location on the screen. This was used initially to understand the icons that may have been tapped, and later for the applications that were launched. Their work tries to take this concept further by identifying individual keystrokes on a virtual keyboard. However, their work has involved a group of 10 volunteers but our work involves inferring typing patterns through an application and predicting the results in real time. Our work collects only around 100 characters per participant in the form of 3 sentences for training as well as testing, as our focus has been to minimize the amount of data as possible to correctly identify a user's keystrokes. Ping et al. [7] attempt to infer long text such as mail messages or tweets. They have used machine learning algorithms to roughly predict the input text and then use linguistic models to further correct the wrong predictions. The accuracy of about 30% after machine learning has significantly improved to 65% after using the linguistic models. However, client applications on smartphones have inherent drawbacks, because these applications cannot be executed unless users download them to their own devices, and these applications can only run on a specific operating system. Thus, some researchers wanted to design web applications to circumvent these problems. More recently Shen et al. [8] have built in the approach taken by TapLogger with taps being recorded with the accelerometer, gyroscope and the magnetometer to infer the tap positions. According to Felt et. Al [9], the common approach is a security model based on permissions, which allows users to perform a relatively complex risk-based security decision in order to access potentially sensitive information or capability (such as location sensors). This complicated model has been difficult to manage for most users. The reason for this complexity can be either due to the unavailability or unpreparedness of the users to completely understand the risk behind giving permissions to an app or because apps are over privileged, meaning they request for extra permissions than that are required to work. Mehrnezhad et al. [10] have proposed PINlogger.js to steal users' privacy when users input PINs on web browsers. However, their one-phase prediction accuracy has not been ideal on some devices because of the sampling rate restriction on motion sensors. WebLogger [11] was designed to overcome this problem and improve the prediction accuracy on web platforms by introducing the weighted voting algorithm in model training phase. All the existing works didn't make use of the language structure or the pattern in the language being typed. Our work aims to use a language model to infer the text. Using the language information, the errors can be handled during the classification and thus results in better accuracy.

III. SYSTEM DESIGN

A. System Architecture

The block diagram of the entire system is shown in the Fig. 1.

An android application for data collection is developed to collect sensor values of accelerometer and gyroscope in the background. The application currently used in our experiment is a basic notepad application. It can be extended to any other application as well. The app also collects key presses while being in the foreground. The sensor data collected along with the key presses forms a labelled data that represents the phone motion while tapping each key. Again, on test data we try to infer the tapped keys from only the sensor readings of Accelerometer and Gyroscope. In our experiment, we begin by identifying taps on the test data using a statistical technique. Then we divide the qwerty keyboard into six regions numbered from 1 to 6. The keyboard regions are shown in the Fig 2.

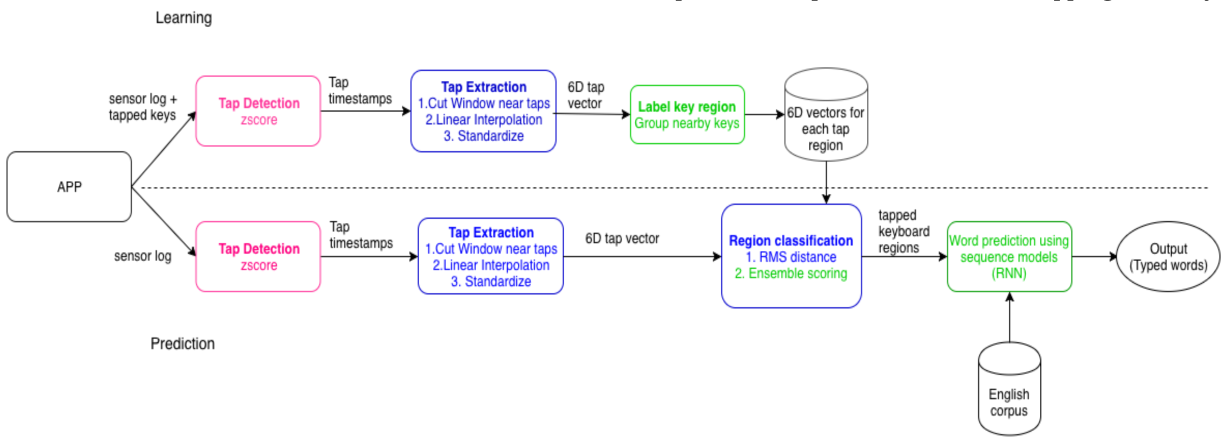

Fig. 1 System Architecture

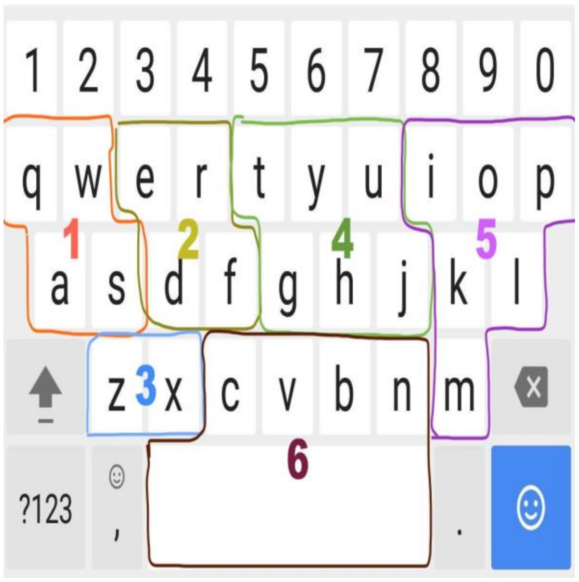

Fig. 2 Keyboard regions

A classifier model is used to classify the test data taps to one of the 6 regions based on the labelled sensor readings. Thus, for each tap, we now know the keyboard region where the tap has happened. After obtaining a sequence of keyboard tap regions for a word, all possible English words matching the region sequence are generated using sequence models such as RNN and language models.

B. UI Design

For data collection purpose, the android application lets the user to record sensor recording on tap of a button with his consent. Once the sensor data recording starts, three randomly chosen sentences are shown to the user. The user is asked to type the sentences and click stop to end the session. The application automatically uploads this to the Firebase data storage. The remaining part of the project uses CLI to process the sensor logs and to view the inferred word suggestions.

C. Methods

Data Collection: When the user types in our android application, it records key presses along with the readings of accelerometer and gyroscope for supervised learning. This facilitates the inference of victims typing in other apps using the sensor readings. This app initially explains the user about the work and gets consent from the user to collect the data. Once the user completes typing, the log files get uploaded automatically to the online firebase storage.

Tap Detection: Given the time series sensor data collected from the application, this module detects the taps on the screen and outputs a list of timestamps where the taps have occurred. Z-score algorithm is one of the most common signal peak detection algorithms. This uses a robust version of Z-score to detect peaks in the signal and to effectively perform an anomaly detection. The anomalies in the signal are taps. The peak signal detection algorithm is made more robust by replacing the mean with the median and also median absolute deviation is considered in place of standard deviation. This has given us profound results with better accuracy than the previous method.

Tap Extraction: Given the timestamp of the taps, the sensor data are windowed near the taps. A window of a predefined millisecond duration cuts out sensor readings starting from a defined time period before a tap and ending at a defined time period after the tap. For each tap we get a 6D vector, (3 accelerometer axis and 3 gyroscope axis). This 6D vector represent the movement of the phone during the corresponding tap. Since this vector is non- uniformly sampled, it is resampled uniformly using linear interpolation. Then the vectors are standardized by subtracting mean and dividing them by the standard deviation. A finalised 6D vector is obtained after interpolation and standardization. This set of 6D vectors obtained from each tap in the training data is labelled with the corresponding tap region and stored.

Region Classification: Given a 6D vector, the closest match is obtained by comparing the given sample with all the available 6D vectors from the reference data. Distance of this 6D vector from all other samples is calculated using the RMS distance. RMSE is a technique used to compare two time series data. An ensemble scoring based method is used with the distance metric to come up with a tap region.

Word Prediction: Given the sequence of keyboard regions as tapped by the user, the probable sentences or words typed are derived by using the knowledge of letter probabilities and the English language. A character is trained on an English corpus to predict the next probable character given a sequence of characters. The RNN outputs the probabilities of all the characters in the vocabulary, to be the next character. The top 3 probable letters that are present in the next keyboard region are selected. They are appended to the current word and thus the finalised words are obtained.

Sentence Prediction with LSTM Approach: Given a sequence of keyboard regions as tapped by the user and a trained model, this module makes use of a predictive keyboard to infer the next probable words typed by using the knowledge of letter probabilities in the English language. A two-layer 128 neuron LSTM network with an additional SoftMax layer is trained on an English corpus to predict the next n probable completions of a word given a sequence of characters. The words are predicted until a space character is obtained The LSTM outputs 10 probable word completions for the character sequence. By using region mapping mechanism, the most probable word can be inferred. This works well for some of the cases.

IV. ANALYSIS AND EXPERIMENTAL RESULTS

The analysis was performed on data obtained from *Samsung Galaxy s6 edge* running Android 7.1. A paragraph of about 3 sentences is typed to create samples for each tap region. Following that, 24 words (10 unique) are typed while holding the phone in the same position. From the samples, the we try to infer the 24 words.

A. Tap Detection

The Accelerometer and Gyroscope sensor logs are shown in the Fig 3 and Fig 4 respectively. The color bars represent the taps.

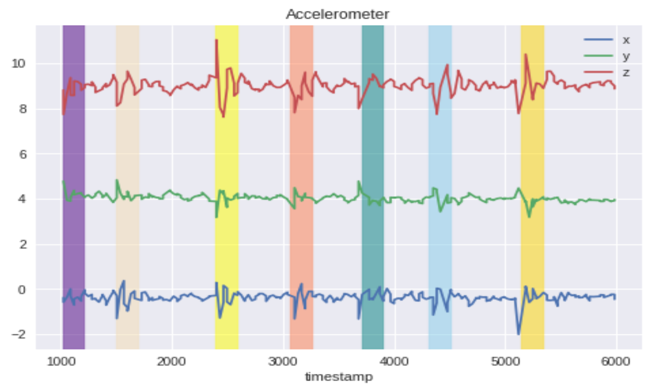

Fig 3. Accelerometer Sensor readings

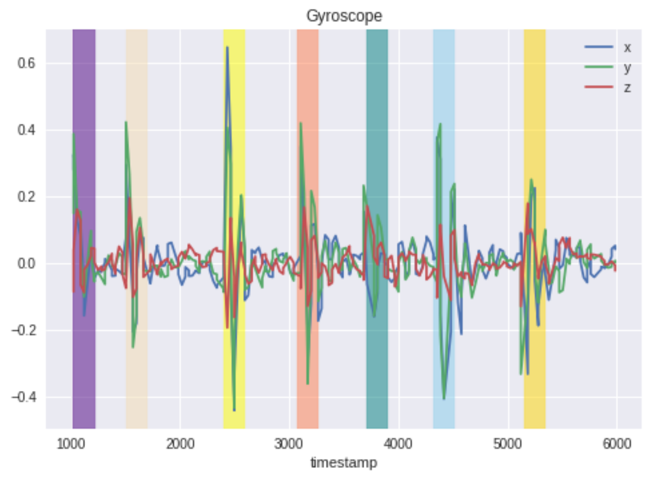

Fig 4. Gyroscope Sensor readings

The deflection in the sensors during tap can be clearly seen and thus a peak detection algorithm is used to detect taps in time series data. One of the two sensors would suffice to perform the task and so accelerometer was chosen.

The magnitude of the 3 values from the accelerometer can be picked or only the z-dimension can be chosen alternatively to obtain similar results. The 3-axis accelerometer readings are converted into a single time series data by calculating the magnitude. This time series data is running through Z-score to detect the taps. The precision of tap detection is calculated by Equation 1. The recall metric of tap detection is calculated by Equation 2.

$$precision = \frac{\text{No. of actual taps detected}}{\text{Total No. of taps detected}} \quad (1)$$

$$recall = \frac{\text{No. of actual taps detected}}{\text{No. of actual taps}} \quad (2)$$

The F1 score is given by Equation 3

$$F1\ score = 2 \times \frac{precision \times recall}{precision + recall} \quad (3)$$

The threshold factor of the Z-score algorithm can be used to balance between the precision and recall. Increasing the threshold will reduce the false positives and also the recall. Reducing the threshold will shoot up the recall and at the same time decrease the precision because of the false positives obtained. The results from Z-score and robust Z-score are compared for further analysis. The robust Z-score gave a f1 score of 0.921 whereas the normal Z-score gave a f1 score of 0.801. The Figure 5 shows the three metrics against the 24 sample test data.

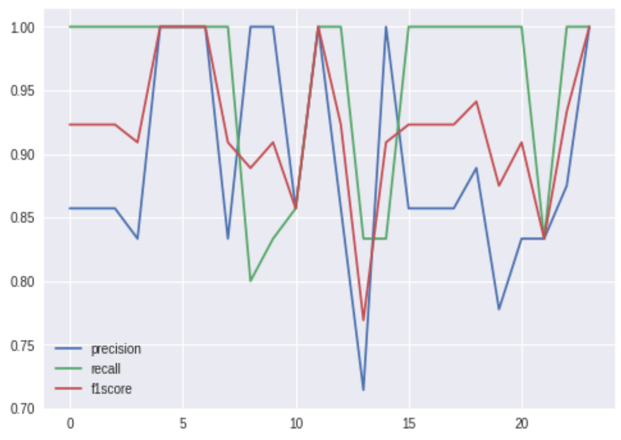

Fig. 5 Tap detection result

From the Figure 5, it can be seen that there is not much consistency across the experiment. Change in hand stress during taps seem to affect the results and thus accounting for the lesser f1 score in certain test cases.

B. Region Classification

6D vectors: From the labelled reference data obtained from data collection, 6D vectors are extracted for each tap and labelled with the tap region. The obtained 6D vector for keyboard region 1 is shown in the Figure 6.

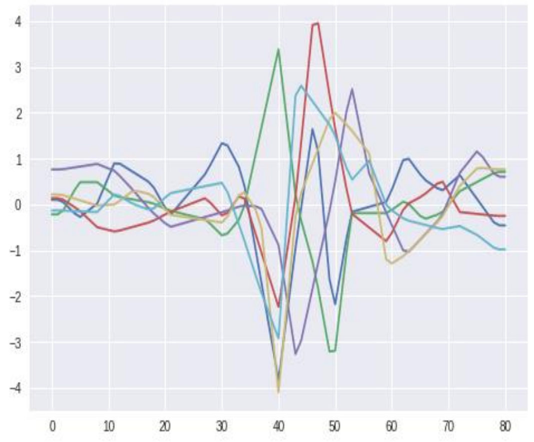
Fig. 6 6D vector - Region 1

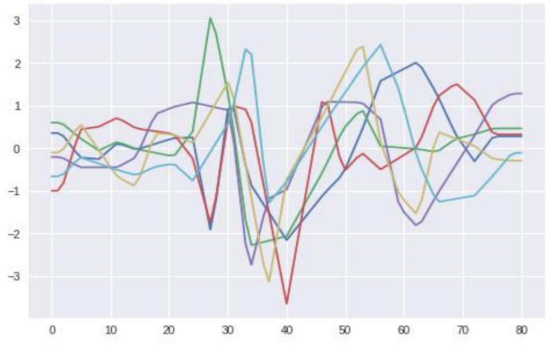
Fig. 7 6D vector - Region 3

It can be seen that the 6D vector differs from the one corresponding to the tap in Region 3 shown in Figure 7. For each keyboard region, more than one reference 6D vectors as shown in Figures 6 and 7 are extracted.

Tap region classification: From the tap timestamps obtained from the tap detection and 6D sample vectors from reference data, each test data tap needs to be classified among one of the 6 keyboard regions. The samples contain 6D vectors for each region representing the movement of the phone during a tap in that region. Initial research started with using common machine learning models such as SVM, Random Forest for the classification task. The 6D vector is under sampled and flattened to form the feature vector which is labelled against the corresponding region. This intuition however seems not suitable as the samples is very less. The reference data is dependent on the user, mobile device and the hand posture. Thus, a more flexible approach is needed.

RMS *(Root mean square)* distance can be used to compare the similarity of two data series. Employing RMS distance by comparing the unseen 6D vector to every sample 6D vector and the taking closest match seemed to work. Since the closest match could be an outlier, a scoring algorithm is introduced to improve the classification. Instead of the closest match, all the distances are sorted and assigned a score based on its position in the sort. The closest gets the highest score. The scores are added grouping by keyboard region. The output class is determined by the region with highest score. By this, even if the closest match is wrong, many correct regions below will add up to handle the outlier. For this scoring to work, all the regions should have equal number of 6D samples, else the classifier will be biased to the region with more samples. The accuracy of the classification is calculated by Equation 4.

$$Accuracy = \frac{No.\ of\ correctly\ classified\ regions}{Total\ No.\ of\ taps} \quad (4)$$

The average accuracy of region classification from the 24 test cases is 0.5612. The Figure 8 compares the metric between scoring and closest match.

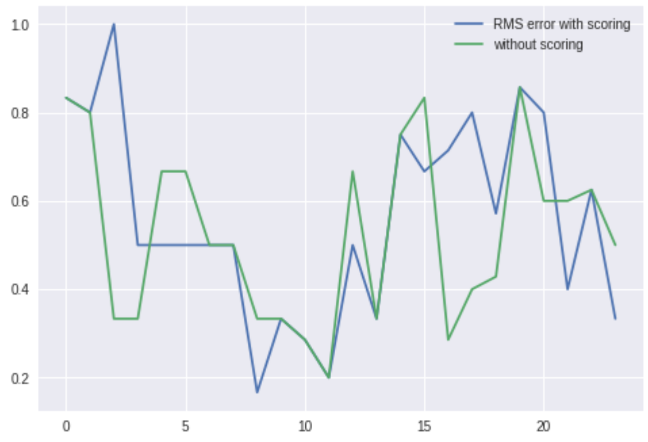

Fig. 8 Tap region classification result

Again, it can be seen that the accuracy isn't consistent among the files. Since the regions are tightly placed in the keyboard, there isn't much difference between the 6D vectors of adjacent regions. In fact, the samples of adjacent regions overlap. This inference of tap region can be serious threat to user privacy. Given the personal details of the user and this keyboard tap regions, the user's password could be inferred. Given this tap regions, the password wordlist for brute force attack can be made much shorter and thus posing a security issue. Also, by employing this technique on the entire screen, the app usage behaviour of the user can also be inferred.

C. Text Inference

Word Inference: From a sequence of keyboard regions tapped, the probable words matching that needs to be generated. The challenge here is that the region sequence itself isn't accurate. A two-layer RNN with 512 GRU each and a third fully connected layer is trained on a corpus[1] of top 10000 English words. The RNN outputs probability for the next character given the current character along with its state. RNN is robustly made to generate all probable words by taking top three possible matches for each character position. The RNN output probabilities for the nth character is filtered by the nth tap region to include only the keys in that region. Given the nth tap region and the probabilities of the keys in it to be the nth character, top three characters are chosen and added to the word. Since the RNN doesn't have state for predicting the first character of the word, All the keys in the first tap region are taken. Repeating this, robust word generation creates a set of 3^{l-1} *x No. of keys in first tap's region words*. Since the word generation is robust and considered 3 possible character for each position, the words that are not close to English are removed. Then the words are autocorrected to account for the errors in the region sequence. The resulting bag of words are the inferred probable words for the given region sequence. This robust generation gave an inference rate of 0.375 (9 out of 24 inferred). This can also be extended to inferring sentences, given that the spaces can be detected with high accuracy.

Region size: The number of keyboard regions can be reduced by having a greater number of keys per region. This increases the accuracy of region classification as the regions become larger and number of target classes is reduced. On the other hand, increasing the number of regions favor the word inference as the number of probable words for a given region sequence is reduced.

The experimental results are shown in Table-I.

Table-I: Metrics

S. No	Technique	Metric
	Tap Detection	**Detection Rate**
1	Standard z-score	0.8017
2	Median z-score	0.9210
	Region Classification	**Accuracy**
3	RMS Distance	0.5212
4	RMS Distance with Scoring	0.5612
	RNN Test Cases	**Avg. Accuracy** on 100 samples
5	While standing	0.375
6	While moving	0.0
7	Phone on table	0.0

https://github.com/first20hours/google-10000-english/blob/master/google-10000-english.txt

V. CONCLUSION AND FUTURE WORK

In our research, we have proposed an approach to identify the user tap positions even if we do not have any large historical sensory data of this particular user. We have obtained an accuracy of 37.5% which proves to be better than the previous methods which have been used in this research. The experimental results demonstrate that our proposed method can predict tap sequences and can generate words matching the sequence.

This research can be extended to infer passwords which would prove to be a potential threat to user's security. The character RNN can be trained on a password wordlist, so that it generates password patterns. Using such an RNN with the knowledge of keyboard tap regions obtained from our tap region classification method, can help generate probable passwords making the password brute-force attack much more feasible. Another extension would be to infer sentences with context using a LSTM model. One major challenge with using such a complex language model is that one wrong inference due to error in tap region sequence propagated through modules, could result in the change of context and hence affect all the corresponding sentences we have to infer.

APPENDIX

A. List of Abbreviations

NLP - Natural Language Processing
CLI - Command Line Interface
UI - User Interface
RMSE - Root Mean Squared Error
RNN - Recurrent Neural Network
LSTM - Long Term Short Memory Network
6D - Six Dimensional

REFERENCES

1. Daniel Genkin, Lev Pachmanov, Itamar Pipman, Eran Tromer, and Yuval Yarom, "Ecdsa key extraction from mobile devices via non-intrusive physical side channels", In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, pp. 1626–1638. ACM, 2016.
2. Liang Cai and Hao Chen, "Touchlogger: Inferring keystrokes on touchscreen from smartphone motion." Hotsec vol. 11, pp. 9–9, 2011.
3. Zhi Xu, Kun Bai, and Sencun Zhu, "Taplogger: Inferring user inputs on smartphone touchscreens using on-board motion sensors", WiSec'12 - Proceedings of the 5th ACM Conference on Security and Privacy in Wireless and Mobile Networks, 04 2012.
4. Adam J Aviv, Benjamin Sapp, Matt Blaze, and Jonathan M Smith, "Practicality of accelerometer side channels on smartphones", In Proceedings of the 28th Annual Computer Security Applications Conference, pp. 41–50. ACM, 2012.
5. Emmanuel Owusu, Jun Han, Sauvik Das, Adrian Perrig, and Joy Zhang, "Accessory: Password inference using accelerometers on smartphones", In Proceedings of the Twelfth Workshop on Mobile Computing Systems &#38; Applications, HotMobile '12, pp. 9:1–9:6, New York, NY, USA, 2012, ACM.
6. Emiliano Miluzzo, Alexander Varshavsky, Suhrid Balakrishnan, and Romit Roy Choudhury, "Tapprints: your finger taps have fingerprints", In Proceedings of the 10th international conference on Mobile systems, applications, and services, pp. 323–336. ACm,2012.
7. Dan Ping, Xin Sun, and Bing Mao, "Textlogger: inferring longer inputs on touch screen using motion sensors", In Proceedings of the 8th ACM Conference on Security & Privacy in Wireless and Mobile Networks, p. 24. ACM, 2015.
8. Chao Shen, Shichao Pei, Zhenyu Yang, and Xiaohong Guan,"Input extraction via motion-sensor behavior analysis on smartphones",Computers & Security , vol. 53, pp. 143–155, 2015.
9. Adrienne Porter Felt, Elizabeth Ha, Serge Egelman, Ariel Haney, Erika Chin, and David Wagner, "Android permissions: User attention, comprehension, and behaviour", In Proceedings of the eighth symposium on usable privacy and security, p. 3. ACM, 2012.
10. Maryam Mehrnezhad, Ehsan Toreini, Siamak F Shahandashti, and Feng Hao, "Stealing pins via mobile sensors: actual risk versus user perception", International Journal of Information Security vol. 17, num. 3, pp. 291–313, 2018.
11. Rui Song, Yubo Song, Qihong Dong, Aiqun Hu, and Shang Gao, "Weblogger: Stealing your personal pins via mobile web application",2017 9th International Conference on Wireless Communications and Signal Processing (WCSP), pp. 1–6, 2017.

AUTHORS PROFILE

Dr.P. Uma Maheswari M.E., M.B.A. Ph.D., is being an Associate Professor, Department of Computer Science and Engineering, College of Engineering, Guindy,Anna University, Chennai. She completed her B.E., degree in the Computer Science and Engineering discipline, M.E., in Software Engineering and PhD in Data Mining from Anna University. During the tenure of service of about 22 years, Dr.P.Uma Maheswari played various roles and known for her contribution leading to value-creation and value-addition in various sectors such as overall administration, academic affairs, education and research. She organized many international, national and regional conferences, workshops, faculty development programs and has delivered 100+ guest lectures, seminars and workshops. Her research interest includes Artificial Intelligence, Machine Learning, Character Recognition from Ancient scripts, Image Processing, Deep Learning, Internet of Things, Natural Language Processing and intelligent computing. Her contribution to the research is well understood through her 93 research publications in reputed research journals with 154 citations with 6 h-index, and 3 i-index, more than 10 technical reports and 8 text books. As a recognized Research supervisor of Anna University, Dr. Uma Maheswari has produced 12 PhD awarded and 4 awaiting thesis evaluation report and doctoral Committee member for more than 30 Ph.D. scholars. She is being a reviewer of many reputed journals and evaluator of ISSRD student projects. She is an active member in professional bodies of ISTE, IEEE and CSI.

Mohamed Yilmaz Ibrahim -A final year Computer Science undergraduate at the College of Engineering Guindy, Anna University, Chennai. He has interned in Cyber Security at the Indian Institute for Development and Research in Banking Technology, Hyderabad as well as in NLP at the L3S Research Centre, Leibniz Universität Hannover, Germany. His research interests include Cyber Security, Machine Learning and Natural Language Processing.

Ramkumar B -A final year Computer Science undergraduate at the College of Engineering Guindy, Anna University. He has interned at Samsung Research & Development Institute Bangalore, India. His research interests include Cyber Security, Android and Machine Learning.

Aswin Sundar -A final year Computer Science undergraduate at the College of Engineering Guindy, Anna University. He has interned in Android App Development at GoBumpr - Northerly Automotive Solutions Private Ltd., Chennai, India. His research interests include Cyber Security, Machine Learning and Android App Development.