Truth Estimation in Form Filling from Mobile Interaction

Negin Yaghoubisharif
116503
negin.yaghoubisharif@
uni-weimar.de

ABSTRACT
Filling the application forms has been mostly transferred from paper to online platforms today. The authenticity of the input information is of great importance especially when it comes to more sensitive information such as applying for a job or visa, and when it comes to preventing social media security issues by taking control of creating accounts information and preventing creating of fake accounts. Besides, with the advent of mobile devices and smartphones, many of these form filling processes are done on people's choice of mobile devices. In this paper we propose a concept of estimating truth or lie of the input data in filling a form on Android mobile phones with the help of the sensor data gathered during interaction of the user with a specific application on their phone. We trained a classifier with the gathered data and achieved the result of weighted average F1 score of .80.

Author Keywords
mobile application; Android application; application UI; machine learning; feature sets;

INTRODUCTION
As many forms and application are filled online using mobile devices everyday, estimating the truthfulness of the data is essential in many cases. As to the study done in [1], a way of estimating the truthfulness of the input on mobile devices is not based on the input content, but on the physiological responses of people to telling a truth or lie.

In this study, we assume that when the user is lying, their physiological responses are different compared to when they are telling the truth; therefore, when people are using mobile devices, these responses would affect on the devices' sensor data, that is, for instance, we assume that when the user is asked a question, it takes longer in case they want to lie rather than telling the truth.

In order to prove the hypothesis, we implemented a voluntarily lying Android application in which the users can choose whether they want to lie or be honest in filling a simple form with their own information; meanwhile, the data is gathered during the users' interaction with the application UI and stored on web server.

In this project, after implementing the Android application, we conducted a study on five participants, each of whom did one test on the application including 10 questions, and gathered the data on server to use them as the dataset for training the classifier. The results from our study showed 50% precision, and 50% recall for truths (F1= .50), and 87% precision and 87% recall for lies (F1= .87), with the weighted average F1 score of .80.

RELATED WORKS
Our project is based on the concept of Veritaps which is proposed in [1]. The paper introduces an additional layer of security by identifying truths and lies on mobile devices.

The truth estimation in Veritaps is independent from the content, and the truth or lie is identified by the data collected from mobile sensors. The hypothesis was that the physiological responses to lying, like increased hand activity can be identified through smartphone sensors while tapping or swiping. To prove that, they conducted three crowdsource smartphone studies to gather the input data.

In first study Simple Lies are identified through an application. In this study the participants did an online experiment on their own phones. The task was to tell the truth or lie about the color of the screen using swipes and taps in "directed" and "choice" conditions whether to lie or tell the truth. Overall, participants were asked to lie and tell the truth half of the time each, and the response was by tapping a button in half of the trials, and swiping on the other half. The results showed that the response time of the lie cases was bigger than that of telling the truth cases.

In second study they implemented an Ultimatum Game in which the participants played with AI as the responder (Mary). The first participant is given an amount of money, then they should propose a division of this money to Mary and state the original amount of money they had using num-pads. The worker would receive points depending on if Mary accepts the deal or not. During the trials the sensors data was collected. For classification a binary (truth/lie) classifier were built. The results showed that honest participants did less hand movement (lower mean

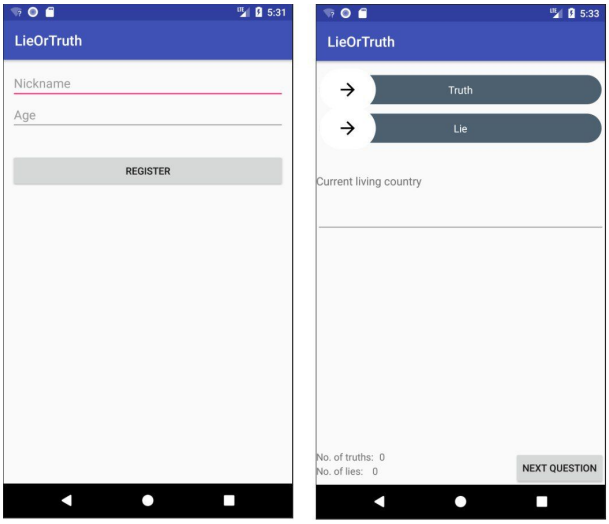

Figure 1. application UI for (left) registration, and (right) one question of the test

acceleration), plus, the duration between first and second event on num-pad was longer for liers.

Study 3- Yatzy Game: Here a dice game with 12 rounds was developed, in each round 5 dice were appeared and the worker had to choose from a list of possible combinations and entered the score that the combination made, which is the sum of the dice. They were rewarded based on the points at the end. The results showed an F1-score of .98 in classifying truths. And higher pressure on taps and longer time on num-pad for in lying cases were detected.

While the original concept of veritaps proposed an optional layer on mobile interaction and determined the validity of results through three games, in this project we focused more on one specific application of this concept and investigated the accuracy of the results on filling forms. That is, our objective here was to add more security to online applications and form filling processes.

Besides, in this work pre-physiological and post-physiological responses to lying are taken into account; the former in the process of decision making, whether the user wants to lie or tell the truth using two swipe buttons of truth and lie and the time of sliding them, and the latter by having the pressure on the button to the next question.

IMPLEMENTATION/STUDY DESIGN

For this study, five participants were asked to install the application on their phone and they were explained briefly about how to work with the application, and they were informed that the input data they enter would be stored on server and used for later analysis..

After installing the application, the first step is registration. As shown in figure 1 (left), the user is asked to register by assigning a nickname and their age in case they had not registered before. The account information is then stored on the server together with a unique id for each user. The server python code is available on github. Having been registered the users are able to do multiple tests with the same user id. A test contains 10 questions. The questions are fetched from database and shown in app in random order, each question is appeared in one page as shown in figure 1 (right). The questions are basic information about the participants as follows:

- Gender
- Number of siblings
- Current job
- Marital status
- Whether they have criminal record
- Current living country
- Nationality
- Height in centimeters
- Their mother's family name
- Eye color

The users are asked for maintaining an equal distribution of lies and truths answering the questions, and they select this by swiping the sliding button "Lie" or the sliding button "Truth" from left to right above each question.

The features we collected is based on four main hypotheses we made.

First, we assumed that there is more hand movement when the user is lying than when telling the truth. For this, the average amount of acceleration sensor data in X, Y, and Z axis together with their magnitude for each question are collected.

Second, lying can affect pre-physiological responses; deciding to lie might take longer than deciding to tell the truth, which is determined by the time the user slide the swipe button Lie or the swipe button Truth from left to right. Besides, when the user swipe one of the buttons the other button gets deactivated to make sure they cannot choose both at the same time. However, they can change their mind by tapping on the selected button to get back at the initial state and select the other option. The swiping time is overridden on the previous number.

Third, lying can also affect post-physiological responses, which were here measured by the pressure when tapping on the button "Next" to bring up the next question. We hypothesize that the pressure is more intense when users are lying.

Fourth, answering a question takes longer when people lie rather than when they tell the truth. This data is gathered for

two different response time features. First, the time that the user is typing the answer in the EditText starting since the user focus on EditText by tapping on it till when they take the focus out of Edittext to another View like the button "Next". When lying, users are more likely to be hesitant about the answer, and while they have tapped on the EditText, they might still need to think more before typing the answer, or they may even change their mind about the answer and remove characters and type again. Second, the time it takes to answer a whole question including deciding to tell the truth or lie, reading the question, thinking, answering the question, and tapping button to the next question. The starting time point here is as soon as the question appears and the end point is when the user press next button.

After a test is done, all the data is stored on the users' phone cache database and on web server. The database contains four tables of User, Test, Questions, and Answer Features. The Answer Feature table contains the features for later classification. Each user has a unique user id and they can do multiple tests, for each test there is a unique id as well. The data is then sent as a json file to server.

CLASSIFICATION

As for the classification, we used Weka software [2], which contains tools for classification. The list of used feature for the classification is as followed:

1. avgSensorX: the average amount of acceleration force along the x axis with normal sample rate for each question
2. avgSensorY: the average amount of acceleration force along the y axis with normal sample rate for each question
3. avgSensorZ: the average amount of acceleration force along the z axis with normal sample rate for each question
4. avgSensorM: the average amount of the magnitude of acceleration force along all three axes with normal sample rate for each question
5. etDuration: the time it takes that the user fill the edit text
6. swipeButtonDuration: the time that the user swipe the Swipe Button Truth or Swipe Button Lie
7. answerTime: the time for answering a whole question; from when the question appears to when the button next is touched
8. btnPressure: the pressure on the button "Next" -from the getPressure method in MotionEvent class- which returns the current pressure of an event; returns more than one number
9. btnPressureListAverage: the average of the float numbers returned by each tap on the button "Next"
10. btnPressureListNo: the number of floats the getPressure() method has returned by each tap on button Next

The features are set in a list of attributes in an arff file and is given to Weka as input dataset. The arff file for this project is available on github. Our dataset contains 50 feature sets; for five participant, each of them did one test with 10 questions including five truths and five lies. Each feature set include all the mentioned feature one participant for one question. We splitted the 80% of feature sets as training set and the rest as test set. We used a couple of classification methods including REPTree. Using REPTree algorithm, the result from our study showed 50% precision, and 50% recall for truths (F1= .50), and 87% precision and 87% recall for lies (F1= .87), with the weighted average F1 score of .80. The classification information is briefed in the confusion matrix.

Issues

Applying this project came with multiple issues, some of which were solved and some required more time. The very first issue was that the time the user was spending time on typing in EditText was affected by the number of characters, for this, we divided the time by the number of characters on EditText. However, the user might type a number as input and depending on their phones' keypad, changing the keypad from alphabet mode to numbers can delay the input time to different extents. This makes it worse when the input is a combination of alphabets and numerics, such as " 29 September 2013" in which the user should change the keypad mode twice. However, in our study the questions are asked in a way that the answers are either only alphabets or only numbers.

The other problem is that if keyboard auto-correct or the dictionary-provided words are activated in the users' phone keypad, and the users take the advantage of them, the time of input would be reduced and does not make sense to take them as an effective feature to train the classifier. Therefore, in this study, we ask the users to deactivate these options or not to use them during doing the test.

CONCLUSION

Today many applications of form filling and account

making are done online, and many people tend to use their mobile phones for these intentions. Plus, the validation of input data is significantly important. In order to evaluate the truthfulness of the input data, we conducted a study based of the concept of Veritaps [1]. The study was done by implementing an Android application of a form with 10 questions appearing one by one. Five participants were asked to install the application and do the test. They had to lie in filling the form for five questions and tell the truth for the other five and determine this by sliding the related swipe button. During the users' interaction with the application UI, the phone's sensor data was gathered and sent as json file to server to provide features sets to train classifier. For the classification, we used REPTree classification method with the help of Weka software. We achieved a rather satisfying result for detecting lie with 87% precision and 87% recall (F1= .87), and weighted average F1 score of .80.

Future work

According to [1], few features are not enough in order to identify the truth of an input, but a list of features is required for more accurate result. As to the future work we recommend training a machine with bigger Dataset to classify truth or lie in form filling with more sensor data as features and see which features are the most determinative in classification. As the users' age could have an effect on their response time and hand movement, we recommend to take age as a feature of the classifier into account, which is why for registering in our application we ask for the users' age. However, as the the participants in our study were the students with the same age range -from 28 to 32 - in our classification it does not make any sense to consider it as an effective feature. Another effective feature for further studies is to consider facial emotion recognition and pupil movement as additional feature to train the machine. Having trained the classifier, the test could be designed in a way that users do not select between truth and lie, but they do it unintentionally with the help of some motives for people to lie in some cases. The swiping buttons of truth and lie, therefore, can be removed from the user interface so that they do not force the user to lie. The use case of sliding button could be in answering a question with two alternatives such as single or married. Other fields of forms can be also added to the user interface such as different types of checkboxes like radio type.

REFERENCES

1. Veritaps: Truth Estimation from Mobile Interaction, Aske Mottelson, Jarrod Knibbe, Denmark, Kasper Hornbæk, Published in Proceeding CHI '18 Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems Paper No. 561 ACM New York, NY, USA 2018
2. https://www.cs.waikato.ac.nz/~ml/weka/index.html

Index

Link to github repository of this project:
https://github.com/Neginysh/MIS-Project

Link to the video of application:
https://github.com/Neginysh/MIS-Project/blob/MyBranch/video.mp4