Birju Patel

Dr. Garrison

March 25, 2024

Project 2 Report

Summary

Authentication requires having a user present something they either know, have or are to prove their identity. Most organizations use password-based authentication, and when additional security is needed, layer on a second factor like a phone number or email. However, these approaches are prone to human error. Users often use the same password for many services, or else use easily guessable passwords such as their birthday. They also often leak passwords to adversaries conducting phishing attacks. Since most users use the same password for every service, the adversary could then potentially log into the user's email, bypassing the second factor. These problems could be avoided with biometric authentication. Though it is the most secure, as it is difficult to fake a person's physical features, it is rarely used because collecting biometric data requires specialized hardware. Users want to authenticate into their services from any machine, but most laptops are not built with fingerprint scanners or retinal cameras.

The speed and cadence with which one types varies from person to person. This data can be easily collected by a background process that monitors the keyboard. My project aims to show that this data can be used for biometric authentication. There are many advantages to this method. Users are already used to typing in passwords, so having them type another standard phrase will not create too much friction. The technique could also be used in the background. Consider an application where users remotely edit documents. By monitoring keystroke patterns, an administrator can verify that the user that was logged in was the one who was actually making the edits, and not an adversary that has broken into their account.

Experimental Setup

To collect keystroke data, I created a typing test which prompted users with a series of phrases and asked them to retype them exactly. As users typed each phrase, their keystrokes were logged by a background process. The program produced a time series for each phrase consisting of a list of keystroke events. Within each event was stored the key's value, whether the key was pressed or released, and what time the event occurred. The program then saved this data for future analysis. In total, 10 participants completed the test, including myself. During the test, each participant typed 50 phrases, 30 of which were unique and 20 of which were repetitions of the same phrase.

I then used this data to train a few machine learning models that predicted the user given unlabeled keystroke data. This model was integrated into an authentication program. The program authenticates users by asking them to retype a phrase three times, and allows them access if the model classifies at least two of them as having been typed by that user.

Model Selection and Evaluation

Machine learning models require a fixed number of features to learn a classification function, but the raw data is a variable length time series. To train the model, I extract the following features from each time series.

- The average time that each key was held down.
- For each possible pair of keys, the average time between when the first key was released and the second key was pressed down.
- The variance in the time each key was held down for the 10 most frequent keys in English (e, t, a, o, i, n, s, r, h, and l).
- The number of times backspace was pressed.

From the keystroke data from a single phrase, these features are used to predict which user generated that data. The first feature, hold down time, approximately captures how much pressure the user applies to their keys when typing. The second feature, wait time between pairs of keys, captures how fast the user moves from one key to the next, which indicates their typing speed and hand size. The final two features, variance of hold down time and backspace count, attempt to capture if a user is bad at typing, as if the user is a poor typist, they likely to type at an erratic pace and make more mistakes. The models consider 3092 features in total.

I use 3 types of models, a random forest classifier, a support vector machine, and a logistic regression. I train these models on 2 datasets, one containing data from all phrases, and the other contains only data from the phrase repeated 20 times. I use grid search to tune hyperparameters. In total, I trained 6 models. Below is a summary of the results of the models.

Classification Results, All Phrases, Name=Birju			
Model Name	Precision	Recall	Accuracy
Random Forest	1	0.22	0.93
SVM	0.5	0.55	0.91
Logistic Regression	0.5	0.44	0.91

Confusion Matrix, All Phrases, Random Forest	Label=Not Birju	Label=Birju
Actual=Not Birju	91	0
Actual=Birju	7	2

Confusion Matrix, All Phrases, SVM	Label=Not Birju	Label=Birju
Actual=Not Birju	86	5
Actual=Birju	4	5

Confusion Matrix, All Phrases, Logistic Regression	Label=Not Birju	Label=Birju
Actual=Not Birju	87	4
Actual=Birju	5	4

Classification Results, Fixed Phrase, Name=Birju			
Model Name	Precision	Recall	Accuracy
Random Forest	1	0.33	0.95
SVM	0.5	1	0.93
Logistic Regression	1	1	1

Confusion Matrix, Fixed Phrase, Random Forest	Label=Not Birju	Label=Birju
Actual=Not Birju	39	0
Actual=Birju	2	1

Confusion Matrix, Fixed Phrase, SVM	Label=Not Birju	Label=Birju
Actual=Not Birju	36	3
Actual=Birju	0	3

Confusion Matrix, Fixed Phrase, Logistic Regression	Label=Not Birju	Label=Birju
Actual=Not Birju	39	0
Actual=Birju	0	3

The best performing model is the logistic regression, followed by the support vector machine and then the random forest classifier. When the models are trained on varied data, they perform relatively poorly. The high accuracy here is misleading, as since the classes are imbalanced, a model can be 90% accurate by simply classifying everything as negative. However, when trained on the data from a fixed phrase, the performance improves drastically, and the logistic regression model classifies these examples perfectly. I suspect that if more data was added, the logistic regression will eventually make mistakes. After retraining the models several times, I was able to find instances of the model misclassifying a few items. Below are the average false negative rates after retraining each model 5 times on the fixed phrase data.

Average False Negative Rate, Fixed Phrase, Name=Birju, Retrained 5 Times	
Random Forest	0.753
SVM	0.107
Logistic Regression	0.073

Average False Positive Rate, Fixed Phrase, Name=Birju, Retrained 5 Times	
Random Forest	0.0052
SVM	0.016
Logistic Regression	0.0216

The logistic regression classifier has a low false negative rate, which means that it has a low probability of rejecting a legitimate user. It also has a low false positive rate, so it will not incorrectly accept an imposter. Therefore, it is convenient enough to use for authentication.

Conclusion

I integrated the logistic regression model into the authentication program, and tested it on 7 of the original 10 participants. In all cases, the model accepted users who tried to log in under their own name, but rejected them when they tried to authenticate under another. Out of the 7 tests, not a single participant was given unauthorized access or was needlessly denied access.

However, while the system worked on a small scale, I have serious doubts that it could be used for authentication in the real world. The classifier can only give a probabilistic estimate that a user is who they say they are, not a guarantee. Even though the final logistic regression model has a low false positive rate of 2.16%, this can still be brute forced after a hundred or so attempts. While this could be partially remedied by using a more sophisticated model trained on more data, the false positive rate on any machine learning model can only be pushed so low. At the same time, an adversary could easily design a robot to do the task automatically. The most practical use case for a system like this would be in fraud detection where it could be used to monitor for suspicious activity, such as flagging if an account has been accessed by an unauthorized user after the breach has occurred.

Future Work

The current model can only differentiate between users if they all type the exact same phrase. Ideally, we would like a model that recognizes the typing behavior of a particular user regardless of what phrase they are typing. Such a model would have to work with arbitrarily long time series data. During this project, I experimented with using a recurrent neural network trained on the raw time series data. I omitted this from my report because the performance of this model was so low. In the future, one could select different features that better reflect the

sequential nature of the typing data. They could then use those features to train an LSTM network, which better models the idea of dependency of future data on past data.

Reflection

During this project, I designed and implemented a piece of privacy enhancing technology. To design the technology, I had to consider the problems with traditional biometric authentication, and proposed a solution that tried to fill the gaps. After implementing the technology, I thoroughly tested it and characterized its strengths, while also honestly considering its weaknesses. Based on this, I recommended what I saw as the right use case for this technology. In conclusion, I believe this project touched on all of my course goals.

Sources

Implementation Of Long Short-Term Memory (LSTM) For User Authentication Based On Keystroke Dynamics, Ferhatovic, Almisreb, Turaev, and Saleh

Scikit-learn Documentation

Notes

To try the authentication tool for yourself, follow these steps.

- Install the python modules pynput and scikit-learn by navigating to the *src* directory and running *pip install -r requirements.txt*.
- Start the program with *python authenticate.py*.
- You must first add yourself as a new user. When you try this, you will be given directions to retype the phrase 20 times. This data will be used to calibrate the model. Type at a normal pace, as you would when you are ordinarily doing work.
- After this, you will be able to log in as yourself. When you try, you will be prompted to type that same phrase 3 times. The program should display a message saying your access has been granted. If you try to log in as someone else, it should say that your access has been denied.