Biometric Keystroke Authentication for Password Entry

Ryan Kearns, Alex Langshur, Henry Mellsop

Stanford University, Spring 2019

Abstract

Through research, we ascertained that typing habits vary subtly between individuals. All together, these habits can be considered a 'biometric signature' of a given person.

In our project, we applied this idea to create an augmented password security system. The key idea is to not only verify whether a password is correct, but to also classify whether or not a given typist is the real user or not, based on this biometric signature.

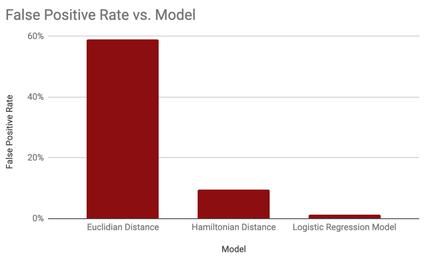
We experimented with various models in an attempt to discover which performed best at determining the authenticity of a user. Our initial attempts relied on Manhattan, Euclidean and Mahalanobis distance inferencing techniques. Manhattan, in particular, performed extremely well.

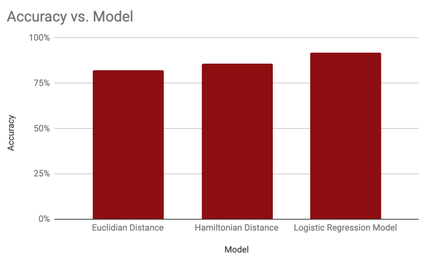
In an attempt to do even better, we applied a logistic regression model over our data, with excellent success, as depicted below. At scale, we believe this system could provide a meaningfully augmented security system – providing a degree of redundancy even when a user's password has been compromised.

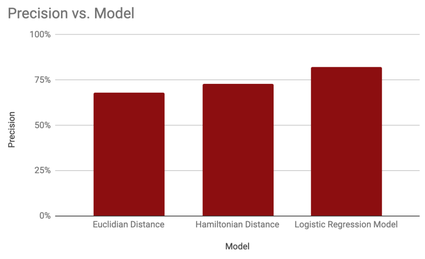
Results

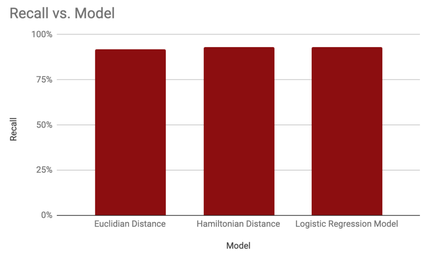
The results of our various modelling attempts are depicted below. As a general outline, our Adam-Optimised logistic regression model was the most successful model by a significant margin. It performed admirably, with an accuracy of 92%, precision of 82%, false positive rate of 1.2% and a recall rate of 93%.

One particular factor to note is how the recall for all models was similar, but how the logistic regression outperforms on other factors. Our model was able to significantly reduce the false positive rate compared to Euclidian and Hamiltonian distance models with identical recall. Put otherwise, the model learned to more effectively bar inauthentic attempts without sacrificing accuracy on genuine attempts. In doing so, our model also improved marginally at both accuracy and precision.









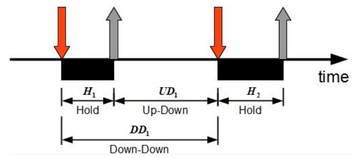
Data Collection

We gathered data from two sources:

- A dataset from Carnegie Mellon University, containing data from 400 individuals, each of whom typed the password ".tie5Roanl" 50 times in carefully controlled conditions.
- Self collected data, using a custom-made keylogger installed onto our own computers using the pynput library. We each collected our own 50 attempts at typing this same password, to compare to the aforementioned dataset.

All our data was parsed into the same format – for each individual password attempt, and for each pair of keys successively pressed, we captured:

- The 'Hold' time – the time from when key *k* is depressed, to when key *k* is released.
- The 'Up-Down' time – the time from when key *k* is released, to when key *k+1* is depressed.
- The 'Down-Down' time – the time from when key *k* is depressed to when key *k+1* is depressed.



Feature Extraction

After we gathered the raw data as outlined above, it becomes important to determine the feature vector $\phi(x^{(i)})$ for each password attempt. Each $x^{(i)}$ is represented as a map with keys $(prevKey, nextKey, e \in \{'UU', 'UD', 'H'\})$ and values as the corresponding time, scored in seconds.

Given that our goal is to create a trainable hyper-ellipsoidal decision boundary, in order to create the feature vector we first:

- Normalize all linear features directly from the raw data, by subtracting the mean of the linear features from the real user, from each data point for every password entry. Denote component *i* from this resultant mean vector as $\widehat{t_i}$. This normalizes all of our data around the origin in *k*-dimensional space, where *k* is the number of linear features that we have in the raw data.
- Create squared features for each linear feature, to represent a variable-radius hyper-ellipsoid.

Bibliography

[1] Andrew Maas, Chris Heather, Chuong (Tom) Do, Relly Brandman, Daphne Koller, and Andrew Ng. 2014. Offering Verified Credentials in Massive Open Online Courses: MOOCs and technology to advance learning and learning research
(Ubiquity symposium). Ubiquity 2014, May, Article 2 (May 2014), 11 pages. DOI: https://doi.org/10.1145/2591684
[2] Kevin S. Killourhy and Roy A. Maxion. "Comparing Anomaly Detectors for Keystroke Dynamics," in Proceedings of the 39th Annual International Conference on Dependable Systems and Networks (DSN-2009), pages 125-134, Estoril,
Lisbon, Portugal, June 29-July 2, 2009. IEEE Computer Society Press, Los Alamitos, California, 2009.
[3] Yunbin Deng and Yu Zhong, "Keystroke Dynamics User Authentication Based on Gaussian Mixture Model and Deep Belief Nets," ISRN Signal Processing, vol. 2013, Article ID 565183, 7 pages, 2013.
https://doi.org/10.1155/2013/565183.

Modelling and Inference Techniques

For our initial modelling attempts, we implemented a variable-threshold, binary interpretation of 'nearest neighbors' using distancing models based on Manhattan, Euclidean and Mahalanobis distances from the real user's mean feature vector to input vectors.

$$f_{\text{Manhattan}}(\vec{x}) = \left|\mu_1 - \phi(\vec{x})_1\right| + \cdots + \left|\mu_{|\phi(\vec{x})|} - \phi(\vec{x})_{|\phi(\vec{x})|}\right|$$

$$f_{\text{Euclidean}}(\vec{x}) = \left\|\vec{\mu} - \phi(\vec{x})\right\|^2$$

These models – particularly Manhattan – were successful, with statistics presented below. We then implemented a more advanced model using logistic regression. Given *n* data points with valid or invalid classifications, we maximize the log-likelihood of our data, as the following:

$$\mathcal{L}((\vec{x}^{(1)}, y), \ldots, (\vec{x}^{(n)}, y); \vec{w}) = \prod_{i=1}^{n} \left[\sigma(\vec{w}^T \vec{x}^{(i)})^{y^{(i)}} \cdot \left(1 - \sigma(\vec{w}^T \vec{x}^{(i)})\right)^{1-y^{(i)}}\right]$$

$$\ln(\mathcal{L}(\ldots; \vec{w})) = \sum_{i=1}^{n} \left[y^{(i)} \ln \sigma(\vec{w}^T \vec{x}^{(i)}) + (1 - y^{(i)}) \ln\left(1 - \sigma(\vec{w}^T \vec{x}^{(i)})\right)\right]$$

$$\vec{w}^* = \arg\max_{\vec{w}} \sum_{i=1}^{n} \left[y^{(i)} \ln \hat{y}^{(i)} + (1 - y^{(i)}) \ln\left(1 - \hat{y}^{(i)}\right)\right]$$

Learning

Learning for the simple models was trivial; we simply computed the mean feature vector for the valid user. The subsequent comparison steps consist of simply comparing the distance between a given example feature and this mean, and seeing if it is smaller than the mean of the distances between all of our known user's data points, and their mean.

For learning with the logistic regression model, we used an Adam-optimized, mini-batch stochastic gradient descent, inspired by the approach used by D. Kingma and J. Ba in their paper "Adam: A Method for Stochastic Optimisation". We perform the following repetitively to train the model:

$g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$ (Get gradients w.r.t. stochastic objective at timestep t)
$m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$ (Update biased first moment estimate)
$v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$ (Update biased second raw moment estimate)
$\widehat{m}_t \leftarrow m_t / (1 - \beta_1^t)$ (Compute bias-corrected first moment estimate)
$\widehat{v}_t \leftarrow v_t / (1 - \beta_2^t)$ (Compute bias-corrected second raw moment estimate)
$\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \widehat{m}_t / (\sqrt{\widehat{v}_t} + \epsilon)$ (Update parameters)

Future Work

We have a number of options for improving our model:

- Adding new features (for example, capturing common typing mistakes, whether the user uses caps lock vs. shift, etc.)
- Implementing a basic neural network to levy our logistic regression models in succession, and learn to combine their values together.

We also have a number of options for improving our data:

- The current demo runs on just over 200 password attempts from three users – we are looking to expand this by collecting more data in addition to the datasets and manual data we've collected from three people. This will enable us to perform more real-world validation. Specifically, we want 50 attempts from 15 other people; we've demonstrated that our model works on data of this scale, and gathering our own data would allow us to perform more nuanced analysis to refine generalizable hyperparameters.