Supervised learning methods for biometric authentication on mobile devices

Valerie Ding
Dept. of Computer Science
Stanford University
`dingv@stanford.edu`

Stephanie Dong
Dept. of Computer Science
Stanford University
`sxdong11@stanford.edu`

Jonathan Li
Dept. of Computer Science
Stanford University
`johnnyli@stanford.edu`

Abstract

We develop fraud detection and user authentication classifiers for mobile keystroke and haptic patterns, achieving 84% accuracy, 90% recall, and 81% precision within one model architecture, and 99% recall and 83% precision across all models. In addition to proposing these models that outperform existing touch dynamics authentication models, we present a secure, space-efficient, and extensible framework for real-time biometric backlogging comparison.

1 Introduction

Keystroke pattern and dynamics classification is an important application of machine learning to computer security and authentication. Much of the existing literature focuses on traditional computer keyboard dynamics analysis, but the massive increase in popularity and computing power of mobile devices in the last ten years has spurred significant interest in biometric-focused authentication models for mobile devices.

Existing literature emphasizes the need for more nuanced security protocols in personal devices. As mobile devices store increasingly valuable and confidential information, learning classifiers to detect fraud is becoming ever more applicable and important. At the same time, a general, space-efficient, and real-time framework is required to be viable in practice. To this end, we develop fraud detection algorithms that use real-time keystroke dynamics data, and propose a space-efficient comparison framework that can be integrated into native software across all mobile devices.

The 2016 Teh *et al.* survey of touch dynamics authentication on mobile devices shows that probabilistic modeling, cluster analysis, decision trees, SVMs, and neural nets are the top most widely used in the decision making process [8]. Sen *et al.* (2014) and Jeanjaitrong *et al.* (2013) used multilayer perceptron, achieving around 80% accuracy on the user verification task for mobile touch dynamics data. However, much of the error was in false rejection and both used datasets of either few subjects or short password sequences [10,11], indicating a need for research in discriminatory classifiers and on larger datasets.

2 Data and Methods

2.1 Dataset and features

We use the MEU-Mobile KSD (Keystroke Dynamics) Data Set from the UCI Machine Learning Repository [1], containing 51 records for each of 56 subjects - 2856 records total - of haptic, momentum, and timing features measured of a common sequence, '.tie5Roanl', typed on a Nexus 7 mobile device. There are 71 features monitored, including Hold, Up-Down, Down-Down, Pressure, Finger-Area, Averages of Hold, Pressure and Finger Area.

Hold .	Hold t	Hold i	Hold e	Hold Shift	Hold 5	Hold Shift
89	92	64	85	123	82	70
90	88	99	83	123	101	81
87	90	83	65	79	73	96
71	81	62	72	83	94	89

Table 1. First five features from first four user typing sequences.

2.2 Interpretation framework

We propose the following comparison framework:

- Initialize user keystroke dynamics profile upon mobile device setup. Store a "typical" user keystroke sequence.
- For each subsequent input keystroke sequence, compare the stored keystroke dynamics profile to the input sequence via a learned classifier to predict if the keystroke dynamics were generated by the original user.

This comparison framework relies on a classifier to detect if two keystroke sequences are from the same user. Such setup allows for reuse of the same classifier to perform user verification on any number of users, by replacing the stored keystroke profile.

In addition, the framework is secure and space-efficient since it stores a minimal amount of original user data, which is already anonymized and can be obfuscated to provide additional security.

2.3 Data preprocessing

To prepare data for training our comparison models, we concatenate each keystroke sequence record with every other record to form a new concatenated comparison vector where the binary label is whether or not the records comes from the same user. This generates on the order of 8 million data points to then feed into our learning algorithms.

We implemented a resampling framework that can utilize a variety of undersampling and oversampling methods to undersample the majority class and oversample the minority class. Resampling ensures parity between labels of different user and same user in the training data. Our resampling framework allows for:

- Oversampling the minority class:
  - Synthetic Minority Oversampling Technique (SMOTE): Generates synthetic data points based on clustering on feature data and assuming continuous features
  - Adaptive Synthetic Sampling Approach (ASAYN): Weighted generation of synthetic data points that prioritizes harder-to-learn examples in the minority class
- Undersampling the majority class:
  - Random Undersampling (RUS): Randomly picks examples from the majority class. Our implementation does so without replacement.

Before training and testing, the data was randomly shuffled and divided 90%-5%-5% into training, validation, and test sets. Additionally, we opted to undersample the majority class to balance the classes in this research.

2.4 Models

We trained logistic models and deep neural network models of varying number of layers, hidden units, and activation functions. The N-layer neural networks we employed had 1 to 10 hidden layers, 10 neurons per hidden layer (ReLU activation), and sigmoid output layer activation (Figure 1). We also experimented with a triangular NN architecture (half the number of units at each successive hidden layer).

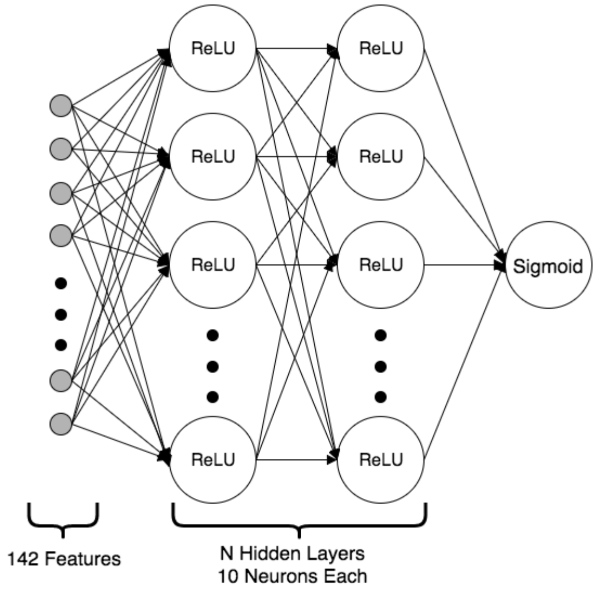

Figure 1: N-layer DNN architecture.

3 Results and Discussion

3.1 Baseline

The baseline compares the first half X_1 of the concatenated vector $X_{1,2}$ with the second half, X_2. For each test example, if $X_1 = X_2$, we label $y_{1,2} = 1$ (same user), and $y_{1,2} = 0$ otherwise. This achieves $\alpha \approx 0\%$ false positive rate, since the only time it predicts $y_{1,2} = 1$ is when the feature sequences are exactly the same, and thus nearly guaranteed, and in this dataset guaranteed, to be from the same user. However, the false negative rate is $\beta = \frac{n-1}{n}$ where n is the number of records per user. With this dataset, this means $\beta = \frac{50}{51} \approx 98\%$. Thus, the baseline achieves 100% sensitivity but 2% specificity.

3.2 Logistic regression with cross entropy loss and no resampling

We trained a logistic regression model optimized using cross entropy loss. For this preliminary stage, we pulled from a subset of the dataset. We generated concatenated vectors for the first 10 examples each of the first 10 users and performed our data preprocessing method on the data subset with no under- or over-sampling. With 70%-30% train-validation split, we achieved 89.6% accuracy with 0% precision, 0% recall, and 100% specificity. Upon inspection, the model consistently predicted the 0 label for every single validation example. We hypothesized that the disproportionate prediction of label 0 was due to heavily unbalanced data, with a significant majority class 0.

3.3 Logistic regression with cross entropy loss and 50-50 undersampling

In our next attempt, we used resampling techniques to balance the majority and minority class to parity. With 50-50 undersampling using random undersampling of the majority class, and using post-processed training set of 260 thousand comparative examples, we achieved 58.12% accuracy, 78.18% precision, 23.88% recall, and 76.12% false negative rate. This model had a 76% likelihood of predicting different user when the user was in fact the same. This is not more effective than a random guess, so the challenge will be lowering the false positive rate, while increasing precision.

3.4 Fully Connected Deep Neural Nets with cross entropy loss and 50-50 undersampling

From the results in the section above, we hypothesized a single logistic unit could not represent enough complexity to capture the relationship between the 142 features of our input. Hence, we trained a variety of fully connect deep neural networks and compared their validation accuracy. Deep neural nets ranging from 1 hidden layer to 10 hidden layers, with 10 neurons per hidden layer, with relu activation, and sigmoid activation on the output layer. The loss function remained cross-entropy. We trained each DNN model for 20 epochs from randomly initialized weights and measured their validation accuracy.

3.5 Triangle NN

To evaluate the effects of varying the number of neurons in the hidden layers, we trained and evaluated a fully connected neural net model with half number of units at each successive hidden layer. This model had 3 hidden layers, 100 neurons in the first hidden layer, 50 in the second one, and 25 in the third one. All hidden units used ReLU activation, and the output neuron had a sigmoid activation. We termed this model the "Triangle" model. We trained this model using cross entropy loss for 20 epochs.

3.6 Summary of results

We compare validation results on all models trained using the 50-50 undersampled data.

Model	Loss	Accuracy	Recall	Precision	False Negative Rate
Logistic	6.661	58.12%	23.88%	78.18%	76.12%
DNN-1	0.690	69.10%	49.05%	**83.31%**	50.95%
DNN-2	7.905	50.41%	**99.78%**	49.76%	**0.22%**
DNN-3	0.536	73.73%	70.37%	76.04%	29.63%
DNN-4	0.695	49.73%	0.21%	59.89%	99.79%
DNN-5	0.531	73.45%	72.52%	74.30%	27.48%
DNN-6	0.409	81.70%	91.18%	77.40%	8.82%
DNN-7	0.484	76.88%	97.16%	69.15%	2.84%
DNN-8	0.527	72.93%	80.23%	70.79%	19.77%
DNN-9	0.425	80.29%	81.69%	80.19%	18.31%
DNN-10	0.450	79.78%	77.46%	81.37%	22.54%
Triangle	0.380	**84.28%**	89.76%	81.14%	10.24%

Table 2. Model results on validation set. False Negative Rate is calculated as 100% - Recall.

From this comparison, we observe that the model with highest accuracy was the Triangle model, with an accuracy of 84.28%. Notably, this model also achieves a high recall of 89.76% and a high precision of 81.14%. For these reasons, we consider the Triangle model best overall. Comparatively, the 2 hidden layer DNN model achieved the highest recall, at 99.78%, and the lowest precision, at 49.76%. Similarly, the 1 hidden layer DNN model achieved the highest precision had a low recall, at 49.05%, as well.

Amongst fully connected NN models with five and fewer hidden layers, the Triangle model accuracy with balanced recall and precision. In addition, as we increase the number of hidden layers beyond 5, the accuracy did not necessary always increase and thus performance on this user verification task was not strongly correlated to the depth of the neural net model. Note that accuracy lowered between DNN-5 to DNN7.

4 Conclusions and future work

We developed and contrasted logistic regression and deep neural network classifiers for user verification through biometric typing pattern data on mobile devices, achieving 84.28% accuracy, 89.76% recall, and 81.14% precision within one model architecture, and 99.78% recall and 83.31% across all models. This outperforms state-of-the-art touch dynamics techniques, specifically neural nets proposed by Sen *et al.* (2014) and Jeanjaitrong *et al.* (2013). We also developed a secure, space-efficient, and extensible framework for real-time biometric backlogging comparison.

Applications of these discriminatory classifiers are in enhancing device security by adding another layer of verification. By writing this classifier onto mobile devices and training it on a user's featurized password input, we can ensure that even if a password's content is typed in properly, it must be typed in with the learned cadence of the original user in order to be verified. This will effectively proof every mobile device from brute force password attacks by adding an unknown number of additional features the attacker must account for. Additionally, this has the ability to continually verify the authenticity of the user based on their typing patterns as they use the mobile device, hardening against device takeover by ensuring that only the primary user has access to the phone.

5 Future work

In this research, we resampled by downsampling the majority class. Our flexible resampling framework allows for different resampling techniques, which can be explored in the future. Additionally, we can perform data augmentation using adversarial examples or other upsampling techniques. This would help inform next-generation development of discriminatory classifiers.

The 83.3% precision of our highest precision model and the 81.14% model of our best overall model is insufficient as a replacement to currently user verification systems, including user ID and password, fingerprint biometrics and face recognition. Future work on this topic would include devising a classifier architecture that would reach acceptably high precision for user verification. Another interesting extension of this research would be to develop a user state model. By modeling each user as a multivariate Gaussian generated from user data and features, we can augment our training data by sampling from user models.

6 Contributions

Valerie Ding worked on data processing, resampling framework, and differential privacy in the authentication models. Stephanie Dong worked on implementing the classification models in Tensorflow and Keras, model training and evaluation. Jonathan Li worked on data processing and algorithm design. All project members contributed to project proposal, milestone, poster and final report.

7 Acknowledgments

We would like to thank Prof. Dan Boneh for motivation behind the research and valuable insight on approaches and application, and Prof. Andrew Ng for guidance in machine learning techniques. We would also like to thank Steve Mussmann, Christopher Sauer, and Alisha Rege for advice and feedback on our research.

8 References

[1] N. Al-Obaidi. MEU-Mobile KSD Data Set. UCI Machine Learning Repository, 2016.

[2] I. de Mendizabal-Vazquez, D. de Santos-Sierra, J. Guerra-Casanova, and C. Sanchez-Avila. Supervised classification methods applied to Keystroke Dynamics through Mobile Devices. *ICCST*, 2014.

[3] T. Cho. Pattern Classification Methods for Keystroke Analysis. *SICE-ICASE*, 2006.

[4] L.J.P. van der Maaten. Accelerating t-SNE using Tree-Based Algorithms. *Journal of Machine Learning Research*, 2014.

[5] A. Fawzi, S. Moosavi-Dezfooli, P. Frossard. Robustness of classifiers: from adversarial to random noise. *NIPS*, 2016.

[6] C. Dwork, A. Roth. Differential privacy. *Foundations and Trends in Computer Science*, 2014.

[7] Y. Gal, Z. Ghahramani. Bayesian Convolutional Neural Networks with Bernoulli Approximate Variational Inference. *arXiv:1506.02158*, 2016.

[8] P.S. Teh, N. Zhang, A.B.J. Teoh, K. Chen. A survey on touch dynamics authentication in mobile devices. *Computers & Security*, 2016.

[9] H. Bae, S. Monti, M. Montano, M.H. Steinberg, T.T. Perls, P. Sebastiani. Learning Bayesian Networks from Correlated Data. *Nature Scientific Reports*, 2016.

[10] S. Sen, K. Muralidharan. Putting "pressure" on mobile authentication. *ICMU*, 2014.

[11] N. Jeanjaitrong, P. Bhattarakosol. Feasibility study on authentication based keystroke dynamic over touch-screen devices. *ISCIT*, 2013.