Project

Papers:

Time–frequency analysis of keystroke dynamics for user authentication

https://librarysearch.le.ac.uk/permalink/f/ns10ce/TN_cdi_gale_infotracacademiconefile_A648519692

Paper from the university of Tehran about how to do it. Has lots of complicated maths such as the winger distribution. Explains their approach in how to do it which has a user enter a test phrase once and then continues to monitor them using time series and the winger distribution to normalise the data. This allows the data to be compared against each other. Difficult to understand the maths. Will need a long hard proper look at. Could I code it? Yes, I think so. Might need some maths related help.

Uses a keystroke based approach and measures how long users spend on each key when typing and can differentiate different users based on that. "The key difference between users is the amount of time spent on each key". Mentions throughout a "cost" matrix and has pseudo code on this. Advocates comparing between too time-frequency representations using 2D correlation coefficient.

Proposed system consists of three general phases: enrolment, authentication and the update phase.

1. Enrolment:
    a. User enters a number of samples as reference
    b. By extracting features of these samples, the users template is constructed.
        i. These samples are generated by recording the users press and release time for each key.
        ii. Then the KDS corresponding to each entry is stored as the reference samples for each user.
        iii. If the input sample includes both the username and password of the user, the KDS extracted from each of them are concatenated to form a single KDS.
2. Authentication:
    a. System takes the input sample and compares this with the users template.
    b. This proposes a way of measuring the similarity perfectly. (Will go into a bit more later on)
3. Update:
    a. The system then decides whether to use the new sample for an update process. (Obviously upon a successful login)
        i. This is needed because as time goes on, users behaviour will change and as a result, may find that they are unable to login.
        ii. Problem is, intruder samples can find their way in and as such intruders may find it way easier to get in. Other problem is, that the program becomes heavier in terms of memory as the number of reference samples increases.
        iii. In order to overcome the first problem, the new sample has to be of a higher similarity. To overcome the second problem, when a new sample is added, the oldest one is discarded which ensures that number of samples stays the same. The paper tries many different approaches to this (FIFO, LFU, and LRU) with FIFO giving the best performance.

The paper then concludes that using this system is better than all ones currently proposed or in use.

NOT CONTINUOUS. Use once on login and thats it.

Keystroke dynamics-based user authentication service for cloud computing

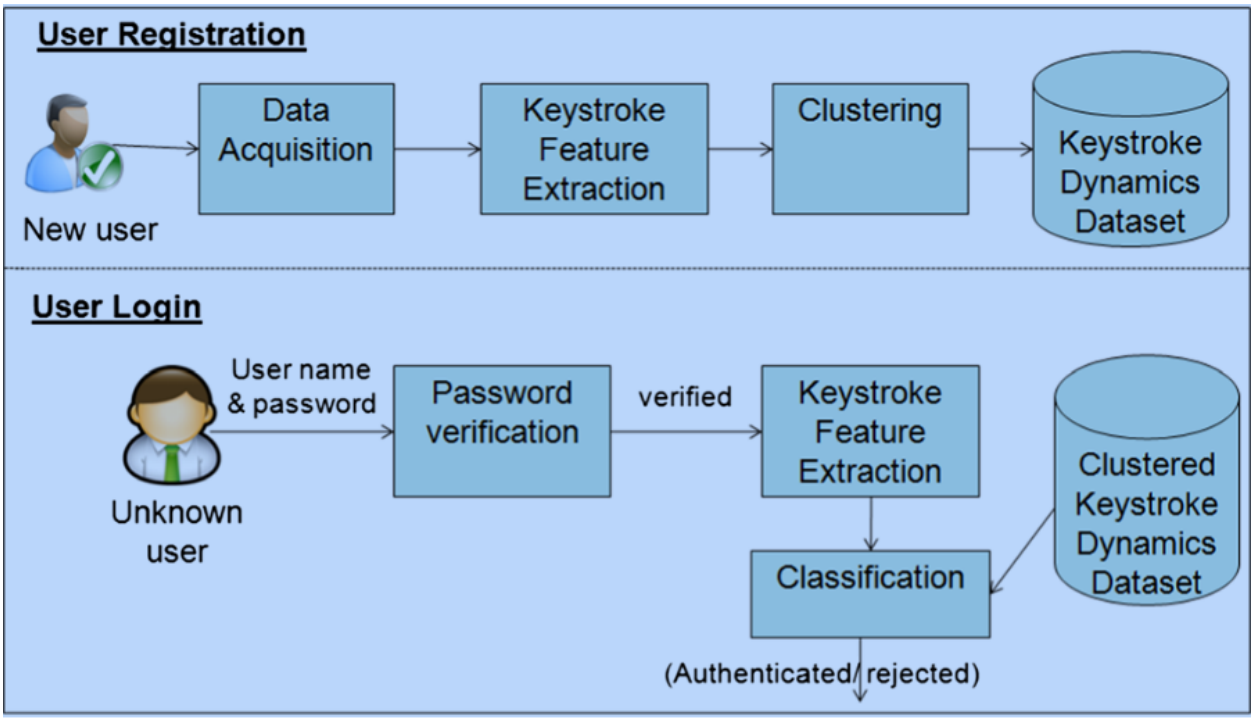
https://librarysearch.le.ac.uk/permalink/f/mvjm1g/TN_cdi_crossref_primary_10_1002_cpe_3718

The paper proposes a keystroke dynamics based authentication system that can detect different users typing. Once again, not continuous.

The proposed keystroke based authentication system.

During registration, the user enters normal login info (Username and PW) along with a number of typing samples. The system then extracts key pieces of info and forms a profile template for the user. The template is then stored in a pre-existing database cluster if one already exists for the user or establishes a new one if it's a new one. In this paper, the template is formed of many different samples gathered over many different sessions. To build the profile template of a user, over 700 features are gathered. The whole keystroke dataset is encrypted.

When the user logs in, the user is prompted for both Uname and PW. If either are incorrect, then they are rejected but if not then the keystrokes are analysed when they entered them in order to verify the user.



Data acquisition:

Gathers raw keystroke measurements are collected via various devices (Keyboard, virtual keyb, touch screen of a phone). In this paper, use datasets (lCMU dataset, GP dataset and the android keystroke dataset)

Keystroke feature extraction

The timing measurement of a user's keystrokes is processed and stored as a profile template for use upon further login. Outlier detection and feature selection methods both take place in pre-processing.

Outlier Detection:

A timing value that deviates from other timing measures in the sample. This might occur when timing hasn't been accurate or an experiment may have not bee run correctly. The system detects outliers using a modified z-score. This is used because the median isn't affected by a few extreme values and as such provides a more robust statistical detection of outliers.

This is as follows:

$M_{ij} = \frac{0.6745(x_{ij} - median(x_{ij})}{MAD_j}$

where MAD_j is the Median Absolute Deviation of feature j, which is the median of absolute deviations from the data's median:

$MAD_j = median(|x_{ij} - median(x_{ij})|)$

The paper suggests that any sample feature with $|M_{ij}| > 3.5$ is considered an outlier.

Feature selection and reduction

Keystroke dynamics suffer from a high dimensionality and a huge number of irrelevant and dependant features which may degrade the classification accuracy and efficiency. Therefore we apply three different feature selection methods.

1. Fishers linear discriminent (FLD)
    a. Machine learning method.
    b. Aims at finding a feature representation by which the within class variance is minimised and the between class variance is maximised. Essentially the aim is to condense high dimensional data to lower dimensions in order to improve verification accuracy and efficiency. For each feature let $\mu_1, \mu_2, \mu_3, ...\mu_m$ represent the statistical means of that feature measurement for user j. FLD score is formulated as follows:

$$FLD = \frac{between - classvariance}{within - classvariance}$$

So the numerator indicates the discrimination between 2 classes, and the denominator indicates the scatter within each class.

$$FLD = \frac{{}^{1}\Big/_{(\mathrm{m}-1)}\sum_{j=1}^{m}\left(\mu_j - \overline{\mu}\right)^2}{{}^{1}\Big/_{m}(n-1)\sum_{j=1}^{m}\sum_{i=1}^{n}\left(x_{ij} - \mu_j\right)^2}$$

where

n = number of features for each user

m = number of different users

x_{ij} = ith feature value for user j

μ_j = mean of all feature values for user j

μ =. mean of all feature values over all users

The FLD score is the highest when a feature exhibits both a low within-class variance and high between-class variance. e.g. the larger the FLD score, the more likely this feature is more discriminative. One disadvantage is that it does not reveal mutual information among features. As such, the proposed system uses feature fusion rules in order to combine the FLD with other feature selection methods.

2. Quickly typed digraphs (QLD)
    a. Implemented as a simple filtering method that considers the close relations between features and their dependancies such as the hold time and the release time of keys.
    b. The QTD method obtains digraphs that are typed quickly, that is having the least typing time. It computes averages of the digraphs for all of the user samples in order to determine the digraphs with the smallest average. Then, the digraph list is ordered ascendingly by the average time. The QTD method consequently selects the top d digraphs as the most QTD and so the most relevant digraphs.
3. Information gain ratio (IG)
    a. Measures how precisely the feature predicts the classes of subject samples, if the only information available is the presence of the feature and the corresponding class distribution. In other words, IG ratio measures the expected reduction in entropy when the features exists vs when is doesn't. Therefore a feature is more relevant if it's normalised information gain is larger. Information gain for feature f is:

$$IG(f) = e(pos, neg) - \left[P_f\, e(TP, FP) + P_{f'} e(TN, FN)\right]$$

where

$$e(x, y) = -\frac{x}{x+y}\log_2\frac{x}{x+y} - \frac{y}{x+y}\log_2\frac{y}{x+y},$$

TP The number of true positives

TN The number of true negatives

FN The number of false negatives

FP The number of false positives

pos number of positive cases = $TP + FN$

neg number of negative cases = $FP + TN$

P_f Probability of feature f

4. Feature fusion methods
    a. The proposed system utilises different feature fusion methods in order to combine all the features extracted from the mentioned features
5. Clustering
    a. In order to improve performance and stop the system searching all records clustering is introduced which stores similar profile templates together in clusters to improve performance.

Classificiation:

A classification method is used in order to determine if the user matches the profile within the system. This system supports and tests three different classifiers: SVM, naive Bayesian and multilayer perceptron (MLP)

a. SVM

    Uses SVM to train and classify the data because SVM can be use to to classify both linearly separable data and nonlinearly separable data. SVM classifies data by finding the best hyperplane that separates all data points of one class from another. The larger the margin is, the better the generalisation error of the classifier will be.

    The nonlinear version of one class SVM algorithm in MATLAB which maps the input data into a high dimensional feature space via a radical basis function kernel:

$$K(x, x') = \exp\left(-\frac{x - x'^2}{2\sigma^2}\right)$$

where x and x^2 are two samples. We set parameters C = 100.0 (the penalisation co-efficient of the SVM) and σ = 1.0 (The parameter of the radical basis function kernel).

b. Naive Bayesian-based classification
    a. Requires a huge of samples to be accurate
    b. This is capable of predicting class membership likelihoods
    c. Very efficient
    d. Let d be a sample that will be classified and et C_y be a hypothesis that d belongs to class y . In the system, we have two classes, genuine and imposter. In the classification process, $P(C_y|d)$ needs to be calculated, which is the probability that the hypothesis C_y holds based on the data sample d. It can be calculated through:

$$P(C_y|d) = \frac{P(d|C_y).P(C_y)}{P(d)}$$

Naive Bayesian assumes that features have independent distributions and thereby estimates:

$$P(C_y|d) = P(d_1|C_y).P(d_2|C_y).P(d_3|C_y)...P(d_n|C_y)$$

c. MLP based classification
    a. Uses this to train and adjust the system with minimal errors to ensure that the system stays up to date.
    b. Neural network
        a. A "feed-forward NN" which means that the information propagates from input to output.
            a. Feed-forwards means that information only goes one way. Forwards. Can also never go backwards.
        b. The network consists of an input layer, several hidden layers and an output layer. The inputs are fed with the values of each feature and the outputs provide the class value.
        c. The hidden layer is an extra layer of neurons with nonlinear activation functions which works as a nonlinear mapping between the input and the output.
    c. This can achieve accurate classification decisions with a small number of samples.

d. The paper advocates using one hidden layer with 25 noes and apply a sigmoid function as an activation function in the hidden layer.

   a. Sigmoid function is a function that is a special form of a logistic function. Denoted by:

$$S(x) = \frac{1}{1 + (exp - x)}$$

      The graph is an s shaped curve.

      As an activation function in a neural network, it will always return either 1 or 0.
      https://machinelearningmastery.com/a-gentle-introduction-to-sigmoid-function/

Results:

This is tested on 3 datasets:

1. CMU dataset: static text keystroke dynamics dataset collected via traditional PC keyboard
2. GP dataset: free text keystroke dynamics dataset collected via a traditional PC keyboard
3. AndroidKeystroke dataset: static text keystroke dynamics dataset collect via a touch screen keyboard

a. CMU datatset

Contains of hold time for each key.

$$HoldTime = KeyReleased - KeyPressed$$

and the latencies of between two successive keys for a static password string which is '.tie5Roanl'. 51 different subjects Total of 400 samples for each user. The data is arranged in a table with 34 columns. Each row consists of one sample.

Evaluation Criteria:

1. FAR: Percentage of imposters let through Defined as $FAR = \frac{FP}{TN+FP}$
2. FRR: Percentage of genuines refused access Defined as $FRR = \frac{FN}{TP+FN}$
3. EER: Value of FRR/FAR at an operating point on ROC where FAR equals FRR
4. Elapsed time of login: Amount of time it take for the system to accept/reject a login.
5. Precision: Percentage of accepted users that are correct genuine users. Defined as $Precision = \frac{TP}{TP+FP}$
6. Recall (Sensitivity): Percentage of genuines correctly accepted Defined as $Recall = \frac{TN}{TN+FP}$

7. Specificity: Percentage of imposters that were rejected. Defined as $Specificity = \frac{TN}{TN+FP}$
8. F-measure: mean of recall and precision Defined as $\frac{2TN}{2TP+FP+FN}$

Conclusion:

Adopting different feature selection methods such as Fisher's linear discriminant, QTD and information gain ration are utilised to reduce the dimensionality of keystroke features. Proposed system also utilises different feature fusion methods to select the most relevant features resulting from the three feature selection methods to improve the authentication accuracy and efficiency. Scalability is used to improve performance.

The system yields very promising results of EER of 0.051 when tested on free text on a pc keyboard. Verification time of 34ms in the case of fixed text.

Not continuous. Only uses at login. Uses machine learning and NN. Further research required into this. Different method from one above. Has a few more steps. Overall, makes sense and a good approach. Could make continuous?? Not as easily as previous. Could use popup every x minutes. Problem is, disruptive to user working. Although, doesn't need to be same text every time. If go with, need to look into deeper.

Keystroke Dynamics-Based Authentication Using Unique Keypad

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/fc290aa2-a206-41de-9c82-e12258d2d0ea/sensors-21-02242.pdf

Software test:

https://www.biometric-solutions.com/keystroke-dynamics.html

No real demos I was able to find. Did take a look into three companies:

- TypingDNA which did one time login but also offer continuous. A test phrase at login to learn and that was it. No info on how they did it. They had an api and the system worked. The demo was lacking and didn't provide much insight on how the actual system would actually work and what measures they took into account. Furthermore, only showed the registration screen and not anything else.

  Tried another one of their products which analysed how the users to type in order to boost their producitivity. Used the same system as above with measuring how users typed. Worked very

well. Could be used as a basis. Was able to measure how the user typed.

- ID Control: Couldn't find any information about this one. Very vague site with no demo or any real information about it. Different site says they provide it however.
- BeahvioSec: No demo. Looked promising but very pricey. They provide a continuous system with an app.
- Currently working through this https://github.com/njanakiev/keystroke-biometrics which is a an attempt using tensorflow and machine learning by two students. Linked to this paper which I'm also reading through currently:http://www.cs.cmu.edu/~maxion/pubs/KillourhyMaxion09.pdf Very interesting. Uses a massive dataset. Only part way through as of rn.