Visualising User Models for Keystroke Dynamics using Empirical Modelling

0713787

Abstract

Keystroke dynamics is a behavioural biometric that hypothesises that an individual can be identified from their habitual typing rhythms. To make authentication algorithms possible, user models are built from timing information extracted from each user typing set phrases of text. Each user model consists of keystroke features, such as the time a certain key is held down for. This paper explores the benefits of applying EM to the field of keystroke dynamics, both as an educational tool and as a visualisation tool. An original model is developed that uses both historic data and live data, input from the user through interaction with the model, to illustrate how user models are built and used in a keystroke dynamics system. From interaction with the visualisation aspect of the model, observations are made that could influence designs of user models in future keystroke dynamics systems.

1 Introduction

The notion of verifying identity using typing signatures can be traced back to the 1850's when it was discovered that telegraph operators could be distinguished from one another by the rhythm with which they tapped morse code (**?**). This discovery, named "The First of the Sender" was used by the military up to World War II to identify messages transmitted by imposters. More recently, the theory has been applied to verifying the identity of people typing on computer keyboards, with applications in computer security.

Computer security, and in particular, identity authentication is crucial for providing secure access to information and system services. The traditional approach to identity authentication is to request a unique identifier, such as a username or email address, followed by an item of knowledge known only to the valid user, such as a password. This method is compromised when the user specific knowledge is written down or shared, and is also susceptible to brute force attacks. This leads to the use of biometrics.

Biometrics are based on unique physical or behavioural traits, such as fingerprints, rather than an item of knowledge that can be mislaid or forgotten. Keystroke dynamics is a behavioural biometric that uses the characteristics of how a user types to identify individuals. The accuracy of keystroke dynamics is not currently strong enough to be used as a standalone authentication method but could be used in combination with traditional passwords to provide multi-factor authentication that rejects obvious imposters based on typing style. A key benefit of keystroke dynamics is that, unlike many other biometrics, it requires no additional specialised hardware. However, the accuracy is affected by temporal variations and factors such as a users physical or mental state.

1.1 Motivations of an EM approach

Keystroke dynamics user models contain a considerably large amount of information, such that it is impossible to visualise what is going on with the data just by looking at the numbers. This paper develops a model that allows the modeller to empirically explore the data and, through interaction with the model, attempt to notice patterns in the user models that could potentially be exploited to enhance the accuracy of a keystroke dynamics security protocol.

The current public knowledge of keystroke dynamics is limited and users may be reluctant to have their typing recorded, especially on security senstive pages such as login pages. An important factor to the successful adoption of a biometric is the user acceptance. For example, users are generally more willing to use a fingerprint scanner than a retina scanner. A keystroke dynamics enhanced login system can be implemented to be completely transparent to the end user as all the work is done behind the scenes and the user does not need to alter their login procedure. While this has benefits for ease of use, it may un-nerve some users to not know what data is being collected and how it is being used. An EM model to explore how keystroke dynamics works could lead to greater understanding and greater user acceptance of the technique.

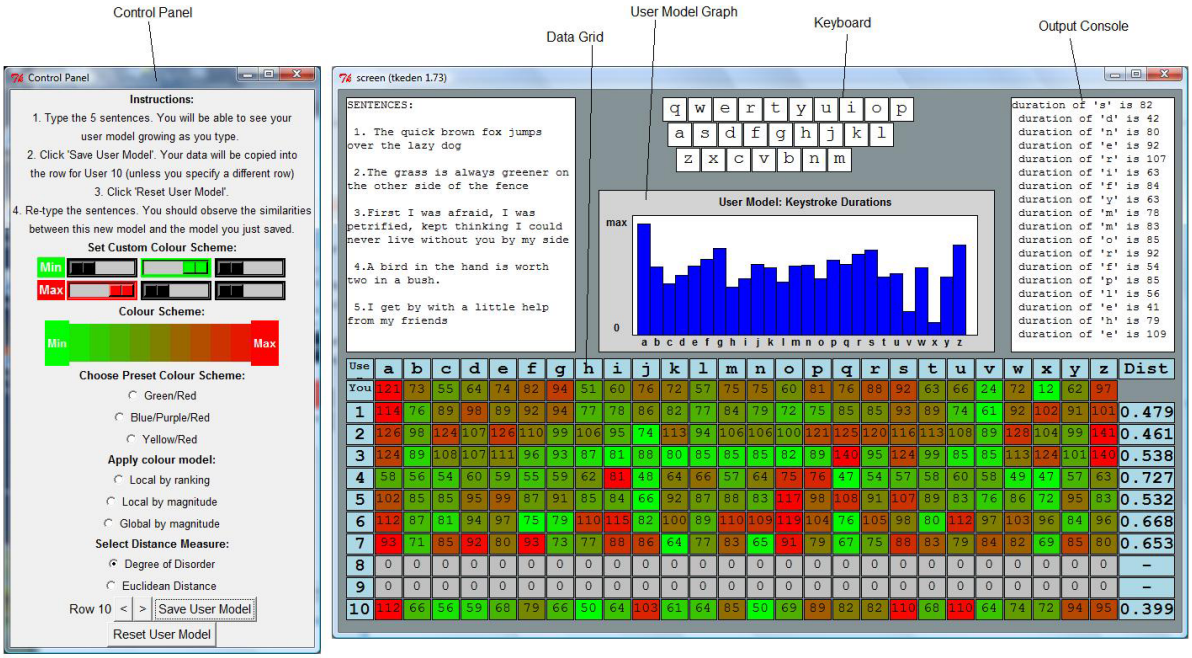

Figure 1: Labelled Keystroke Dynamics Model

2 The KD Model

This section describes the main features of the Keystroke Dynamics (KD) model, shown in Figure **??**. The model is influenced by the sudokuColourHarfield2007 model, which in turn utilises aspects of sudokuKing2006 and gelHarfield2006. The model uses many different EM technologies, including:

- EDEN, the Engine for DEfinitive Notations
- SCOUT, Definitive notation for SCreen LayOUT
- EDDI, EDEN Database Definition Interpreter
- %angel, ANt's prototype GEL, a definition-based graphical environment language for EDEN.

2.1 Interaction with the model

As mentioned previously, the model was designed to consider both the use as an education tool and as a visualisation tool. This section discusses how the model can be used for each scenario.

2.1.1 As an educational tool

Initially, the model is loaded with an empty user model. As the user types on the screen, measurements of keystroke durations are taken and entered into the user model. Each measurement is displayed to the user in the output console at the top right of the screen and the user can visually see their user model updating in bar graph form, in the centre of the screen. This transparency allows the user to see and learn exactly what data is being collected, and how it is being used to form a user model.

The data grid at the bottom of the screen shows the keystroke duration values for each character in up to 11 user models, with the top row corresponding to the live user model currently being typed. By default, the following 7 rows are filled with real data, from 7 sample users, when the EM model is loaded. As the user types and populates their user model, they can observe how the calculated distance between their model and each existing model changes. The user can also save their current model to one of the 10 rows, reset their user model and begin typing again to see how similar the user models they produce are.

For example, in Figure 1, a model was trained by typing the 5 sentences on the screen and then saved to row 10. The live user model was reset and retrained from scratch by the same user typing the 5 sentences again. Of the 8 sample models now stored, the closest

match (smallest distance score) to the live user model was indeed the model originally produced by the user.

Learning is attainable through observation, interaction and redefinition of observables. This enhances the users understanding of how the user models can be applied to calculate distance measurements between models, and ultimately how user models are used in authentication algorithms.

? discusses desirable attributes for an educational model and states that an "empirical model should be accessible, intelligible and correct", highlighting the importance of Human Computer Interaction (HCI) characteristics to a model's accessibility. Originally the KD model was designed purely as a visualisation tool and not an educational model. That legacy is reflected in a some design aspects of the model, which affects the user friendliness for an education user. Most notably, when the user types, no text appears on the screen. This is very unintuitive compared to the standards of HCI where users expect text to appear in a text box. There is also little guidance for an educational user on how to use the model without referring to extra documentation. For example, the user has to know how to interpret distance scores to understand that the lowest distance score is the closest matching user model.

2.1.2 As a visualisation tool

The main interest for a modeller lies in gaining knowledge through observations of the data grid. Each square in the data grid corresponds to a keystroke feature. The modeller can use the colour of the square to infer knowledge on the relationships between different keystroke features. For example, in the default colour scheme, a square is red if it is held down longer than most other keys, and green if it is held down for less time on average compared to other keys. This is similar to being able to infer the state of a square in the sudokucolourHarfield2007 model. You can also use the colouring to infer knowledge across different user models. If the colouring between two user models (rows) is similar, the distance between the user models is likely to be smaller, suggesting greater similiarity in the typing styles.

Interaction is managed through the control panel. From the control panel the colour scheme can be changed dynamically using sliders to help find a scheme that will allow the modeller to notice patterns that another colour scheme would not. The colour scheme specifies a range of colours between 2 chosen colours where the minimum value in a data set corresponds to one colour and the maximum value corresponds to the second colour. The colour scheme can be applied locally (per user model) or globally (across all user models) to help identify different patterns in the data. Similarly, the colour scheme can be applied to the magnitude or to the ranking of elements in a data set. Using the small set of [60, 70, 150] as an example; by magnitude, the colour of 70 will be much more similar to the colour of 60, than 150. However, by ranking, the set becomes [1, 2, 3] and the colour of the 2nd element is equi-distant to both the first and third element. Different distance measures can also be used to explore empirically the distribution of distance scores calculated between different user models.

2.2 Aspects of the model

The EM model centres around measurements and observations of keystroke duration features. Each user model contains 26 features, one relating to the average measured value for each character of the alphabet. This section explores the EM model in more detail, considering the benefits to the theme of visualising user models of each component that makes up the model. Further technical implementation details of each component can be found in the documentation that accompanies the model.

2.2.1 User Model Graph

Although the measurement of each keystroke duration is recorded behind the scenes, the user model graph displays just the average value of each keystroke duration feature in bar graph form. Modelling with dependancy means that the graph updates continually as the user types and more measurements are included in the calculation of average duration for each character. It also means that if the raw data was pre-processed to remove outliers, as is common before use in keystroke dynamics authentication algorithms, the user model would update correspondingly. The bar graph provides a visual way of acknowledging the difference in magnitude of keystroke features, within the live user model.

2.2.2 Keyboard

When a user holds down a key on their keyboard, the corresponding key on the on-screen keyboard changes colour to show the user that the keypress is being recorded. To re-inforce this, the character on the keyboard is replaced with the timing measurement which increases the longer a key is held down for. In reality, a key is rarely held down long enough to observe this timing. When the key is released it

turns white on the screen again. Combined with the user model graph, this allows a user to visually see the effect on their user model if they hold down any key for a particularly long or short time. If the user holds down a key for longer than 2 seconds they will notice that the timing of that keypress is not included in their user model. This is because keypresses exceding a certain duration are not representative of users' habitual typing rhythms. Subtle observations such as these illustrate the benefits of the EM model to users attempting to understanding the workings of keystroke dynamics algorithms.

2.2.3 Output Console

The output console provides transparency to the end user to the data recorded by a keystroke dynamics system. The exact measurement of each keystroke duration is displayed here. The output console is also used to confirm actions such as saving a user model, or resetting the user model.

2.2.4 Control Panel

The control panel uses the prototype GEL notation, %angel, defined by Anthony Harfield, to produce a GUI that allows a modeller to control various aspects of the model previously described. Each component in the GUI typically changes the value of an EDEN observable which through dependencies changes the main model.

While the nature of EM tends to leave the possibilities for interaction deliberately open, the control panel simply nurtures an empirical approach to learning that influences the user's experience whether they are a modeller, educational, or any other type, of user. As well as interaction through the control panel, the user can still also interact with the observables of the EM model. For example, changing measurement values in the user models, or even redefining whol functions, such as the function that calculates the distance between two user models.

2.2.5 Data Grid

The data grid displays the keystroke features in each user model. In addition to the live user model of the user, there is space for up to 10 user models, all of which can be overwritten by live user models if desired. By default, the first 7 rows are occupied with historic models from previous research. These values are loaded from an EDDI database. The use of EDDI is regrettably minimal in the final model, though there is great scope for further work to extend the use of EDDI for querying and sorting data when analysing the user models to look for patterns.

3 Evaluation

The aim of the visualisation aspect of the model is to identify patterns of features that will help distinguish between users. To this end the model has been successful, with a number of patterns able to be observed.

When the grid colouring is set to 'global', it is clear that certain users have higher average durations than other users, not just over some keys, but over all keys. For example, in the default data, User 4 has low average keystroke durations and User 2 has high average durations.This suggests an overall, slower typing speed which would be a useful characteristic to use to distinguish between users.

When the colouring is set to 'local', and in particular when viewed with the ranking of features, it is apparent that some characters have a higher average duration for all users. For example, the character 'a' is consistently one of the keys that is held down longest on average for all users. This knowledge could be used to weight features that are better at distinguishing different users than others. It could also be used to inform clustering of features.

Through interaction of the model it can be observed that, when using euclidean distance, as the number of features in the user model increases, the distance scores also increase. This suggests that degree of disorder is a more reliable distance measure, as the distance score it reports is normalised so that it is always in range 0 to 1 regardless of the number of features used in the distance calculation. In a real system a user model will often be compared to models with different numbers of common features, making normalisation an important feature.

3.1 Limitations

The accuracy of the user models built are limited by the amount of data typed. The suggested five sentences are not enough to train an accurate user model. Ideally, for any learning technique, the number of samples used for training models should be significantly larger than the test set.

There are also inaccuracies in the duration measurements. Often measurements appear to group around the same values. For example, there might be lots of durations around 45ms, and lots of durations being measured as 62ms, but not values inbetween.

This was experienced previously in JavaScript implementations of keystroke dynamics where the measurements were sometimes only precise to approximately 8ms. It has been shown by **?** that the accuracy of keystroke dynamic classification algorithms were robust to imprecision of a few milliseconds, but suffered when the precision is less than 8ms as it becomes impossible to distinguish which keys are being held for longer, on average, than others. This precision is believed to be related to the hardware of the user's machine.

The user is given freedom to type as much as they like, but the data collected is restricted to the 26 characters of the alphabet. In a real system punctuation and formatting (space, delete, backspace) keys would also be used, which would increase the accuracy of the user models.

Another limitation of the user modelling is that only keystroke durations are considered. This is a relatively small amount of data to notice patterns in and to disinguish between users. The other main feature used in keystroke dynamics is keystroke latencies (the time between one key being released and the next being pressed). This feature has a higher dimensionality (n^2, where n is the number of keys monitored) and has previously been found to be a better feature for discriminating between users (**?**).

The KD model currently only displays up to 11 user models at once. Patterns in the data may become more apparent once more user models are viewed synchronously. A potential issue with greater data is that **?** notes that the attributeexplorerRoe2000 model always ran slow when there was a large amount of data ($>$150 records). Extending the KD model using Cadence may provide a solution that works on larger data sets.

4 Conclusion

It has been shown that Empirical Modelling can be used to produce an educational tool that will help enhance the understanding of keystroke dynamics. The model supports discovery in constructionist approach, as discussed by **?** in relation to the sudokuexperienceBeynon2009 model. Users can achieve new knowledge through either following disciplined guidance (by following the suggested interactions), or gain knowledge through their own experiences with the model (by actively exploring and experimenting individually). Through interacting with the model normally and exploiting the underlying observables it is possible for users to construe how changes affect the user models and the resulting distances between user models.

For visualisation, the model can be used to notice patterns that would be impossible to notice in raw data and hard to spot without exploiting the agency and dependancies afforded by EM.

4.1 Future work

There is much scope for further work on the model and a few ideas are outlined below:

- Use EDDI to to load and save user model data to a database. This would make it possible to retain user models between sessions.
- Allow the values in the data grid to be sorted in ascending or descending order by each header (e.g. by a specific character, or by the distance scores). This would provide the potential to notice a greater number of patterns in the data.
- Provide an actual textbox to type text into so that the user can see what they are typing. This would more accurately model a real keystroke dynamics application. The user could also be presented with randomly chosen sentences to type.

Acknowledgements

The author would like to acknowledge the contributions of Dr. Sarabjot Singh to the work on keystroke dynamics at the University of Warwick. The author would also like to thank Dr. Meurig Beynon for his guidance, support and enthusiasm to this project. Additionally, the author would like to thank all the volunteers who have provided typing data to make such studies possible.

References

0408961. Can empirical modelling facilitate learning?: Modelling graphs to assist in the teaching of ai search algorithms. *WEB-EM4*, 2008.

Meurig Beynon and Antony Harfield. Constructionism through construal by computer. *Constructionism 2010, Paris*, 2010.

Matthew Carter. Online user authentication using keystroke dynamics. *CS310: Computer Science Project Report*, 2010.

Matthew Carter and Sarabjot Singh Anand. Building user models for keystroke dynamics on random sample text. 2010.

John. C. Checco. Keystroke dynamics and corporate security. 2003.

Chris Roe. Model readme file. *attributeexplorerRoe2000*, 2000.