K

Kernel

A kernel k is a function that for all $\mathbf{x}, \mathbf{z} \in \mathbb{X}$: satisfies $k(\mathbf{x}, \mathbf{z}) = \Phi(\mathbf{x}) \cdot \Phi(\mathbf{z})$, where Φ is a mapping from the input space $\mathbb{X}$ to the feature space $\mathcal{H}$, i.e., $\Phi : x \mapsto \Phi(x) \in \mathcal{H}$. A kernel function can also be characterized as follows: Let $\mathbb{X}$ be the input space. A function k: $\mathbb{X} \times \mathbb{X}$: $\mapsto \mathbb{R}$ (or $\mathbb{C}$) is kernel *if and only if* for any $M \in \mathbb{N}$ and any finite data set $\{\mathbf{x}_1, \cdots, \mathbf{x}_M\} \subset \mathbb{X}$, the associated Gram matrix is positive semi-definite.

▶ Non-linear Techniques for Dimension Reduction

Key Binding

▶ Biometric and User Data, Binding of

Keypoints

▶ Local Image Features

Keystroke Dynamics

▶ Keystroke Recognition

Keystroke Pattern Classification

▶ Keystroke Recognition

Keystroke Recognition

Nick Bartlow
West Virginia University, Morgantown, WV, USA

Synonyms

Behavioral biometrics; Keystroke dynamics; Keystroke pattern classification

Definition

Keystroke recognition is a ▶behavioral biometric which utilizes the unique manner in which a person types to verify the identity of an individual. Typing patterns are predominantly extracted from computer keyboards, but the information can potentially be gathered from any input device having traditional keys with tactile response (i.e., cellular phones, PDA's, etc). Although other measurements are conceivable, patterns used in keystroke dynamics are derived mainly from the two events that make up a keystroke: the Key-Down and Key-Up. The Key-Down event takes place at the initial depression of a key and the Key-Up occurs at the subsequent release of that key. Various unique features are then calculated based on the intra-key and inter-key timing variations between these events. After feature extraction, a wide range of algorithms can be employed to establish whether the unique pattern confirms or denies the claimed identity.

Introduction

The earliest form of keystroke recognition emerged in the early 1900s during the days of World War I. During the war, the French used listening posts in which operators were able to recognize the "fist" of enemy radio operators communicating in Morse code. These trained individuals would learn to recognize operators by differing lengths of pauses, dots and slashes, and varying transmission speeds. This intelligence subsequently allowed the French to establish the identity of entities such as enemy battalions. Far more sophisticated than electromechanical telegraphs used to transmit Morse code, keyboards of today offer many more opportunities to establish the unique manner in which one types. Intuitively, coarse level differentiation can be achieved by investigating typing speeds. For instance, a professional typist who averages 90 or more words per minute would be easily distinguished from a "hunt and peck" amateur who averages only 20–25 words per minute. That said, this feature only goes so far as many people type at similar speeds and the average speed that an individual types can vary significantly depending on many factors. The time it takes an individual to locate a key (sometimes referred to as "seek-time") also varies from key to key. For instance, left-handed individuals may have quicker seek-times for keys on the left side of the keyboard and vice versa [1]. Along those same lines, use of the shift keys to modify characters can also vary from individual based on handedness and typing skill. Trained professionals will always modify characters on the right side of the keyboard with the left shift key while amateurs may continually use the right shift key to do so [2]. Language undoubtedly plays a large role in the individuality of a typing signature. Given that a person speaks English, commonly used words like {the, and, you, are} are often "programmed" in one's mind and typed quickly as opposed to an individual of a different native language. Additionally, individuals typically exhibit a consistent pattern of errors including replacements, reversals, and extraneous hits. In an extreme case, the consistent lack of errors is a pattern in itself.

Keyboard Technology and Semantics

There are four different kinds of switch technology used in keyboards today; pure mechanical, foam element, rubber dome, and membrane [3]. Each switch type has various characteristics such as feel, durability, price, etc. No matter the key switch technology chosen, when a key is depressed, a degree of "bounce" is present. Bounce can be defined as the effect when the contact device rapidly engages and disengages over an extremely short period of time [3]. Keyboards, either external to desktop PCs or internal to laptops and other devices are computers in their own right as they contain a microprocessor, RAM, and sometimes ROM. Using their processors and controllers, they filter out the difference between bounce and two successive keystrokes. Each stroke therefore consists of two events, when the plates are engaged and when the engagement is released or disengaged. Scan codes resulting from these events are sent from the controller in the keyboard to the event handler in the BIOS of the device in question (usually a PC) [3]. Scan codes are recorded by the processor based on a matrix composed of all the keys on the keyboard. The keyboard matrix operates on a buffer that allows for the processing of simultaneous keystroke events. As mentioned before, when a key is pressed down, the plates become engaged. It is at this point that the keyboard processor sends a "make code" encoded as a hex value to the device. The make code can be thought of as including both the key engaged and various other state flags indicating if/how the key was modified by any of the various control keys such as shift, alt, etc. Once the key disengages, a corresponding "break code" is sent to the PC [3]. These ideas form the basis of keyboard technology at its lowest-level.

Using this background as a foundation, the upper level semantics of keyboard operation can be defined. The basis of all features included in keystroke recognition is founded on the keystroke event and the associated make code or break code correlation described previously. Instead of dealing with terms like "make code," "disengagement," etc., researchers usually yield to the more intuitive, higher level definitions below.

1. *Key-down.* The event that fires when a key is pressed down. This corresponds to the event of the keyboard processor sending the device (usually a PC) a "make code." It should be noted that this event will continually fire until the key being depressed is released. The speed at which the Key-down event fires while a key is depressed is referred as the "repeat rate." This is a user customizable property in virtually all operating systems.

2. *Key-up.* The event that fires when a currently depressed key is subsequently released.
3. *Keystroke.* The combination of an initial Key-down event and the corresponding Key-up event.
4. *Hold time.* The length of time between an initial Key-down event and the corresponding Key-up event. Hold time is sometimes referred to as "dwell time."
5. *Delay.* The length of time between two successive keystrokes. It should be noted that this time can be positive or negative (overlapping strokes). Some works refer to delay as "latency" or "flight."

Some highly specialized keyboards can record other information such as the pressure of key strikes, but the foundation of the technology is based on the events defined above.

Feature Representation and Classification

A wide variety of algorithmic approaches have been explored as suitable candidates for the task of keystroke recognition. The problem of keystroke recognition fits well within the general fields of pattern recognition and machine learning; the two main tasks involved in solving problems within these fields are to define the representation of the feature space and the algorithm used to predict the class of samples. As mentioned in previous sections, the features in keystroke recognition are primarily derived from the elements that make up a keystroke. Most algorithms utilize first order statistics such as minimum, maximum, mean, median, and standard deviation of hold times and latencies [2, 4–8] for feature representation. Here, hold times are for individual keys whereas latencies are measured between two keystrokes often defined as "digraphs." Using these statistics, one can either calculate fixed length feature vectors as outlined in [2] or variable length feature vectors as outline in [9]. Fixed length or static size feature vectors will always have a predetermined length despite the length of the input sequence. The size of variable length or of dynamic feature vectors will depend on the size of the input sequence. Although the vast majority of keystroke recognition systems rely on single key hold times and digraph latencies, some approaches define other feature sets including trigraph durations, ordering of keystrokes (when shift-key modification is required), etc. [9].

Beyond feature representation, a keystroke recognition system must employ an algorithm to predict the class of incoming samples. In general, the approaches can be broken down into two sections: distance metric based approaches and machine learning approaches. After calculating the feature vector for an incoming sample, the chosen algorithm must predict the class of the sample (genuine or imposter). Many approaches will do so by comparing the incoming sample to one or more reference samples in a template database through a distance metric. Popular distance metrics include: Euclidean, Mahalanobis, Manhattan, Chebyshev, and Hamming. When distance metrics are employed to compare two samples, the smaller the score the closer the two samples are to each other. Gaines and Lisowski [4], Garcia [10], Young and Hammon [11], and Joyce and Gupta [5] are all examples of algorithms that utilize one or more of these distance metrics as classification schemes. Table 1 provides an overview of selected work in keystroke recognition including the works listed above. The table includes the features/algorithm used, input requirements, the scope, and performance. Under the performance column the raw totals in terms of FAR and FRR are presented within parentheses when listed in the work.

As the field has matured, many other machine learning approaches have emerged as viable solutions for prediction mechanisms in keystroke recognition. Neural networks have widely been employed with works by Obaidat et al. [6, 7], Brown et al. [12], and Maisuria et al. [13]. Cho and Yu have applied Support Vector Machines (SVM's) to the problem extensively [14, 15]. Additionally, Bartlow and Cukic explored the decision tree approach of Random Forests [2] (see Table 1 for more information on listed works).

Applications and Challenges

In application, the uses of keystroke recognition can range anywhere from stand-alone biometric systems to augmenting general computer security systems. Depending on various system specific security characteristics such as database size and operational risks, keystroke recognition is suitable as a stand-alone biometric. Although not on the level of physiological biometrics such as iris, fingerprint, and face, many works in the literature indicate that the attainable performance rates are within the scope of what some operational profiles would require. Much like the physiological biometrics,

Keystroke Recognition. Table 1 Overview of Selected Works in Keystroke Recognition

Work	Feature(s)/Algorithm	Input	Scope	Performance
Gaines and Lisowski (1980) [4]	Latency between 87 lowercase digraphs using sample t-tests	300–400 word passage 2 times	7 secretaries	FAR 0% (0/55) FRR 4% (2/55)
Garcia (1986) [10]	Latency between 87 lowercase digraphs and space key and complex discrimination using Mahalanobis distance function	Individual's name and 1000 common words 10 times each	(N/A)	FAR 0.01% (N/A) FRR 50% (N/A)
Young and Hammon (1989) [11]	Plurality of features including: digraph latencies, time to enter selected number of keystrokes and common words using Euclidean distance	(N/A)	(N/A)	(N/A)
Joyce and Gupta (1990) [5]	Digraph latencies between reference strings using mean and standard deviation of latency distance vectors	Username, password, first name, last name 8 times each	33 users of varying ability	FAR 0.25% (2/810) FRR 16.36% (27/165)
Brown and Rogers (1993) [12]	Latencies and hold times using Euclidean distance and neural networks	Usernames, 15–16 character avg. ≈ 1,000 sequences tested	21 and V 25 users	FAR 4.2%–11.5% (N/A) FRR (N/A)
Obaidat and Macchiarolo (1993) [6]	Digraph latencies between reference strings using neural networks	15 character phrase 20 times each	6 users	97% overall accuracy
Obaidat and Sadoun (1997) [7]	Digraph latencies and key hold times using multiple machine learning algorithms	Username 225 times/day for 8 weeks	15 users	FAR 0% (N/A) FRR 0% (N/A)
Monrose and Rubin (1997) [1]	Latencies and durations with normalized Euclidean distance and weighted/nonweighted maximum probability	Passages of text over 7 weeks	(N/A)	Identification framework
Maisuria and Ong and Lai (1999) [13]	Digraph latencies with neural networks (multi-layer perceptron)	passwords 60 times over 3 periods	20 users	FAR ≈ 30% (N/A) FRR ≈ 15% (N/A)
Monrose, Weiter, and Wetzel (2001) [8]	Digraph latencies and key hold times, algorithm employed is unclear	8 character password	20 users	FAR % (N/A) FRR 45% (N/A)
Bergadano, Gunetti, and Picardi (2002) [9]	Trigraph duration using degree of disorder	683 character text 5 times	44 users	FAR 0.04% (1/10,000) FRR 4% (N/A)
Yu and Cho (2004) [14]	GA-SVM's and wrapper FSS on hold times and digraph intervals	6–10 character passwords 150–400 gen/user and 75 imp	21 users	FAR 0% (N/A) FRR 3.69% (N/A)
Bartlow and Cukic (2006) [2]	Random Forests on digraph latencies and hold times digraph latencies	usernames + 8 and 12 char passwords ≈ 9,000 sequences	41 users	FAR 2% (N/A) FRR 2% (N/A)
Sung and Cho (2006) [15]	GA-SVM's and wrapper FSS on hold times and digraph intervals	6–10 character passwords 150–400/user and 75 imposter	21 users	FAR 3.85% (N/A) FRR 13.10% (N/A)

performance is typically measured by conventional error measures such as False Accept Rate (FAR), False Reject Rate (FRR), and Equal Error Rate (EER). In terms of EER, many of the previously cited works achieve performance $\leq$5% (see Table 1). Naturally, FAR and FRR's can be tailored based on where one wishes to fall on a traditional Receiver Operating Characteristic (ROC) curve. It is important to note that the literature

has not firmly established whether the technology is sufficient for biometric systems operating in identification mode as the focus of past research is almost exclusively tailored to verification based systems. It is also important to note the trend of decreasing data requirements as earlier works required extremely long passages of text whereas most recent works require only usernames, passwords, or both. Related to this trend, keystroke dynamics need not be applied only at the time of login, which may lead to time-of-check-time-of-use vulnerabilities. Instead, they can be applied transparently throughout the span of a period of use. This feature can allow systems to continually check for the presence of insider threat where an authorized user may login to a system and subsequently allow an unauthorized user access. If a system does not require a continual verification environment, keystroke recognition is also very suitable for a ► challenge response type framework where the user is periodically authenticated.

Besides stand-alone biometric systems, keystroke recognition can be used as an augment to traditional username/password systems. This process is often called ► credential hardening or password hardening. Monrose et al. first proposed the idea [8] and Bartlow et al. also explored the concept [2]. Both works show how the addition of keystroke recognition to traditional authentication mechanisms can drastically reduce the penetration rate of these systems. Works of this nature may also bode well in online authentication environments such as banking and e-commerce websites which now commonly require secondary verification layers.

Either as a stand-alone biometric or an augment to a traditional username/password scheme, keystroke dynamics are arguably more cancelable or replaceable than physiological biometrics. The idea of cancellable biometrics touches on the fact that the threat of biometric compromise exists and is often realized. With fingerprint, face, iris, etc., it is often difficult to reissue a biometric authentication mechanism as fingers, faces, and irises are not easily removed and replaced in humans. In keystroke recognition however, the behavior which induces the biometric can be changed. In other words, if a user's keystroke recognition template is compromised, the data in which the template is based (i.e., password/passphrase) can simply be changed which will result in a new biometric template. For obvious reasons, this is seen as a very attractive feature of keystroke recognition.

Beyond the scope of academic research, many patents have been issued in the field including: Garcia (4,621,334 - 1986) [10], Young and Hammon (4,805,222 - 1989) [11], Brown and Rogers (5,557,686 - 1996), and Bender and Postley (7,206,938 - 2007). In addition to patents, there are many commercial offerings of keystroke recognition systems. Two popular systems are BioPassword ©(http://www.biopassword.com/) and iMagic Software ©(http://www.imagicsoftware.com). Systems such as these are attractive as the overhead of keystroke recognition in terms of hardware deployment and seamless integration into currently existing authentication systems is typically much less than that associated with physiological biometrics such as fingerprint, iris, and face.

Despite the maturity of the field over the last 30 years, there are still many challenges that are yet to be solved. Three main challenges are associated with the data required to train keystroke recognition systems. First, few works have formally set out to determine the amount of sequences required to sufficiently establish a typing signature ready for operational deployment. For a system to be deployable, it must have a realistic training requirement that the users are willing to incur. It seems that repeatedly typing a username and password combination 50 or more times would be unacceptable in the eyes of most users, yet five may be insufficient in terms of meeting established security goals. Second, as passwords need to be replaced or reissued, the problem of retraining needs to be addressed. Once again, these retraining requirements are yet to be firmly established. Third, the behavioral nature of this keystroke recognition requires a slightly more involved data collection process than what is typical in conventional physiological biometric systems. Most notably, one cannot simply compare genuine input of one user to genuine input of another user in order to establish an instance of imposter input as the data is often different for every user (i.e., usernames/passwords). As a result, most academic research will have users type the credentials or data associated with other users to arrive at imposter sequences for training. Clearly this is not feasible in operational systems as passwords are frequently reset. Therefore, the issue of automatic generation of imposter data is an area that needs to be explored.

Summary

Keystroke recognition is a behavioral biometric which authenticates an individual not on the basis of what is typed but the nature of how it is typed. A large base of research has accumulated in the field over the last 30 years establishing its potential both as a stand-alone biometric and an augment to traditional username/password authentication schemes. Due to its transparent nature, low cost of deployment, and seamless fit into currently existing commercial and governmental applications, it is an excellent candidate for increasing the security of authentication systems.

Related Entries

▶ Biometric Encryption
▶ Cancelable Biometrics
▶ Verification

References

1. Monrose, F., Rubin, A.D.: Authentication via Keystroke Dynamics. In: ACM Conference on Computer and Communications Security, pp. 48–56 (1997)
2. Bartlow, N., Cukic, B.: Evaluating the Reliability of Credential Hardening through Keystroke Dynamics. In: ISSRE, IEEE Computer Society, Washington, DC, USA, pp. 117–126 (2006)
3. Mueller, S.: Upgrading and Repairing PCs, 15th edn. QUE, Indianapolis, IN (2004)
4. Gaines, R., Lisowksi, W., Press, W., Shapiro, S.: Authentication by keystroke timing: Some preliminary results. Rand Report R-256-NSF, The Rand Corporation, Santa Monica, CA (1980)
5. Joyce, R., Gupta, G.: Identity authentication based on keystroke latencies. Commun. ACM **33**(2), 168–176 (1990)
6. Obaidat, M.S., Macchairolo, D.T.: An on-line neural network system for computer access security. IEEE Trans. Industrial Electronics **40**(2), 235–241 (1993)
7. Obaidat, M.S., Sadoun, B.: Verification of computer users using keystroke dynamics. IEEE Trans. Syst. Man Cybern. **27**(2), 261–269 (1997)
8. Monrose, F., Reiter, M.K., Wetzel, S.: Password hardening based on keystroke dynamics. Int. J. Inf. Sec. **1**(2), 69–83 (2002)
9. Bergadano, F., Gunetti, D., Picardi, C.: User authentication through keystroke dynamics. ACM Trans. Inf. Syst. Secur. **5**(4), 367–397 (2002)
10. Garcia, J.: Personal identification apparatus. Patent 4,621,334, US Patent and Trademark Office, Washington, DC (1986)
11. Young, J., Hammon, R.: Method and apparatus for verifying an individuals identity. Patent 4,805,222, US Patent and Trademark Office, Washington, DC (1989)
12. Brown, M., Rogers, S.J.: User identification via keystroke characteristics of typed names using neural networks. Int. J. Man Mach. Stud. **39**(6), 999–1014 (1993). DOI http://dx.doi.org/10.1006/imms.1993.1092
13. Maisuria, L.K., Ong, C.S., Lai, W.K.: A comparison of artificial neural networks and cluster analysis for typing biometrics authentication. In: International Joint Conference on Neural Networks (IJCNN), vol. 5, pp. 3295–3299 (1999)
14. Yu, E., Cho, S.: Keystroke dynamics identity verification - its problems and practical solutions. Comput. Secur. **23**(5), 428–440 (2004)
15. Sung, K.S., Cho, S.: GA SVM wrapper ensemble for keystroke dynamics authentication. In: ICB, Springer-Berlin-Hiedelberg, pp. 654–660 (2006)

Kinematic Body Model

Virtual skeleton structure comprising a fixed number of joints with specified angular degrees-of-freedom. The values assigned to these joint angles define the 3D pose of the body.

▶ Markerless 3D Human Motion Capture from Images

Kinematics

The description of object motion over time, generally expressed in terms of position, velocity, and acceleration.

▶ Human Detection and Tracking

Knowledge-based Gait Recognition

▶ Gait Recognition, Model-Based

Known Traveler

▶ Registered Traveler