Cost-Effective Kernel Ridge Regression Implementation for Keystroke-Based Active Authentication System

Pei-Yuan Wu, Chi-Chen Fang, J. Morris Chang, and S. Y. Kung

***Abstract*—In this work, a fast kernel ridge regression (KRR) learning algorithm is adopted with $O(N)$ training cost for large-scale active authentication system. A truncated Gaussian radial basis function (TRBF) kernel is also implemented to provide better cost-performance trade-off. The fast-KRR algorithm along with the TRBF kernel offers computational advantages over the traditional support vector machine (SVM) with Gaussian-RBF kernel while preserving the error rate performance. Experimental results validate the cost-effectiveness of the developed authentication system. In numbers, the fast-KRR learning model achieves an equal error rate (EER) of 1.39% with $O(N)$ training time, while SVM with the RBF kernel shows an EER of 1.41% with $O(N^2)$ training time.**

I. Introduction

The present user name and password authentication system has many potential weaknesses [1], [2] such as password disclosure, easy-to-crack passwords, dictionary attacks, etc. The one-time log-in authentication system is also vulnerable to session hijacking, where an impostor may gain access to system resources by obtaining authenticated open sessions that are not properly monitored. Active authentication provides constant non-intrusive authentication by continuously monitoring user-specific physiological [3]–[5] and behavioral [6], [7] biometrics. The physiological features include face [8], [9], retinal or iris patterns [10], [11], fingerprints [12], palm topology [13], gait [14], [15], hand geometry, wrist veins and thermal images, etc. The behavioral features include voice-prints, handwritten signatures, keystroke dynamics, etc.

Physiological features in general have lower error rates than behavioral features, since physiological features do not vary along time as behavioral features do. However, special tools such as iris scanner or video cameras are required to extract such physiological features. This limits the applicability of such techniques due to the increased-cost as well as the lack of current infrastructure. Keystroke dynamics, on the other hand, can be unobtrusively collected using a standard keyboard.

Keystroke dynamics is a behavioral biometric, by which users can be distinguished by analyzing their typing rhythms on a keyboard. Scientists have noticed that neuron-physiological factors involved in handwritten signatures also produce unique keystroke patterns [16], [17]. However, keystroke timing information shows strong variability which depends on the environment as well as the human physiological and psychological conditions.

The study of monitoring keystroke dynamics as an additional layer of protection to the traditional password system has remained active since 1980's [2]. In the earlier work, researchers focused on predefined and structured typing samples, also referred to as *fixed-text* analysis. Fixed-text analysis is mainly used for static authentication during the login stage as password hardening. However, it is not suitable for continuous authentication, since it is unrealistic and intrusive to enforce users to type-in the predefined strings repeatedly throughout the session.

Since the late 1990's, *free-text* analysis has drawn many researchers' attention, which aims to recognize users by the text they freely typed in their daily interaction with the computer. The free-text analysis is suitable for continuous authentication since the data can be collected continuously and unobtrusively throughout the session. Furthermore, free-text analysis allows the user profile to be adaptively refined by continuously collecting the keystroke patterns from users' daily task. However, the unstructured and sparse nature of the information conveyed by keystroke timing data is always a challenge in free-text analysis.

In this paper, we introduce kernel methods into large-scale free-text active authentication system. The learning and prediction system is developed based on a free-text keystroke dataset collected from approximately 2000 participants, which is the largest to the best of our knowledge. Kernel methods are well established in various supervised and unsupervised learning problems [18]–[22]. The basic idea behind the kernel learning approach is to nonlinearly transform the training vectors in the original space onto a high-dimensional intrinsic space [23], characterized by its dimension J, named as the intrinsic degree. Thereafter, various existing linear learning and prediction models can be directly applied to the intrinsic training vectors. If the learning algorithm meets the Mercer's condition [24], or the so-called learning subspace property [23], then the algorithm can be elegantly mapped to the empirical space [23]. This is known as the "kernel trick".

In large-scale authentication system, the data size N tends to become enormously large, rendering it extremely costly to perform kernel-based learning and prediction algorithms in the empirical space. For example, the complexity of conducting machine learning in the empirical space will be respectively in the order of $\Omega(N^2)$ for support vector machine (SVM) [19], [21], [23] and of $O(N^3)$ for the kernel ridge regression (KRR) [25], [26] learning models. This implies a very heavy computational burden to retain the adoption of the (default) Gaussian radial basis function (RBF) kernel. In contrast, if the intrinsic degree may be tuned to a reasonable level such that $J \ll N$, then it will become much more cost effective to perform kernel learning in the intrinsic space, as opposed to the empirical space [27].

In this manuscript we apply the efficient kernel learning algorithm proposed by Kung and Wu [27] to large-scale active authentication system. By approximating the well known RBF kernel with truncated-RBF (TRBF) kernel, the original KRR problem is approximated by a linear least-squares regression problem in the finite-dimensional kernel-induced feature space of TRBF kernel to speed up both training and prediction times.

The remainder of this paper is organized as follows: Section II is devoted to literature survey. In Section III we describe the features collected that serve as the cognitive factors in keystroke dynamics, as well as the authentication system architecture. In Section IV we describe the kernel-based learning algorithms applied in the authentication system, namely the SVM and KRR algorithms. In Section V we introduce the concept of TRBF kernel as an approximation of the Gaussian-RBF kernel, as well as a fast-KRR learning and prediction algorithm. In Section VI a classifier fusion method is described to augment votes from multiple classifiers into final decision. The experimental results based on a large-scale free-text keystroke dataset is provided in Section VII. The discussions and conclusions are summarized in Section VIII.

II. Related Work

A. Fixed-Text Analysis

In Obaidat and Sadoun's work [28], they compared the performance of various pattern recognition algorithms for login string keystroke detection, including fuzzy ARTMAP, radial basis function networks, learning vector quantization, neural network paradigms, back-propagation with sigmoid transfer function, hybrid sum-of-products, potential function, Bayes' rule, etc. Though a best misclassification error of 0% is reported using certain pattern recognition paradigms, it is questionable regarding the statistical significance of their results in large-scale authentication systems, since their study only involves 15 participants.

In Bergadano et. al.'s work [29], 4% false reject rate (FRR) and 0.01% false alarm rate (FAR) was reported based on the keystroke patterns from 154 individuals, each typing a fixed-text of 683 characters for five times. For each typing string sample, the trigraphs within are ordered according to their time durations. They then define a distance measure between two typing samples based on the degree of disorder between their trigraph orderings. A new string sample is classified as belonging to the legitimate user whose known samples have the smallest average distance.

In Sheng et al's work [30], a 9.62% FRR and 0.88% FAR was reported based on a dataset of 43 users, each typing a fixed string of 37 characters for nine times. To attain sufficient training samples, they apply Monte Carlo approach to synthesize training samples by perturbing the existing training samples with Gaussian distribution. They then split the raw and synthetic training samples into multiple subsets, where the monograph and digraph features are extracted to train eight parallel decision trees for each legitimate user. The decision is then based on majority vote.

In Hosseinzadeh and Krishnan's work [31], they combined the keystroke latency feature with Gaussian mixture model (GMM)-based verification system. In their work, each of the 41 participants uses his own full name as the authentication string, and an equal error rate (EER) of 4.4% was reported.

In Killourhy and Maxion's work [32], they collected keystroke data from 51 participants typing 400 passwords each, and then implemented and evaluated 14 detectors from the past keystroke-dynamics and pattern-recognition literature. The three top-performing detectors in their work achieve EER between 9.6% and 10.2%. Their results constitute an excellent benchmark for comparing detectors and measuring process in fixed-text analysis literature.

B. Free-Text Analysis

An excellent literature survey on free-text analysis literature can be found in Alsultan and Warwick's article [33]. Monrose and Rubin's work [34] was among the earliest on the free-text keystroke detection. They collected typing samples from 42 users over a period of seven weeks in various computing environments. For each user, the means of various digraphs are computed to form a user profile. The identity of an unknown user is then classified as the legal user whose profile, as represented by a vector of digraph means, has the smallest Euclidean distance. To reduce the search time in the recognition process, they clustered the legal users' profiles using a maxi-mini-distance algorithm, with their typing speed as the clustering criteria. This however poses an obvious limitation that re-clustering is needed whenever new legal user profile is added or modified. An accuracy of 90% is reported for fixed-text detection, but only 23% for free-text detection.

In Ahmed and Traore's work [35], each legitimate user has a profile of two neural networks that store the monograph and digraph time duration information. In recognition phase, a new user's monograph and digraph time intervals are extracted, which are then compared to the corresponding values predicted by the neural networks of the claimed identity's profile. They collected typing samples from 53 users over a period of five months, and reported an EER of 2.46%.

Gunetti and Picardi [6] extended Bergadano et. al.'s work [29] into free-text keystroke authentication. Based on a free-text keystroke dataset of 205 participants, an EER of 1% was reported. Despite the very low error rates, the computational costs for identifying users were expensive since the test sample is compared to all typing samples from all users in the database. In their experiment, it takes about 140 seconds to compare a new sample against 40 user profiles each containing 14 typing samples on a Pentium IV at 2.5 GHz. Furthermore, the authentication depends not only on the legal user in query, but also on other legal users. These limit its scalability in large networks.

Villani et. al. [36] investigated the case of using different keyboards (desktop and laptop) as well as different context modes (fixed-text and free-text). There were a total of 118 participants. For fixed-text mode each participant copied a predefined text of approximately 650 keystrokes for at least five times; for free-text mode each participant typed five arbitrary emails of at least 650 keystrokes. The extracted features include the averages and standard deviations of key press duration times as well as digraph latencies. They also consider percentages of key presses of special keys. Those features are concatenated into a vector, by which a Euclidean distance criteria is used to compare the extracted features between participants for identification purposes. They acquired 99.5% identification accuracy among 36 users, and 93.3% on a larger population of 93 users, as long as the users stick to the same keyboard and context mode. It was found in their study that the identification accuracy decreases drastically when the users use different context modes or keyboards in the training and testing phases. Furthermore, they found free-text context results in a decreased accuracy as compared to the fixed-text context.

C. Discussion

It appears that except the work by Gunetti and Picardi [6] and Villani et. al. [36], most of the previous text analysis schemes proposed in literature are based on datasets with limited scales, mainly less than 60 participants [37]–[45]. From an algorithmic and system architecture design point of view, a data set collected from several tens of participants may be sufficient. In real world applications, however, an authentication system can easily grow beyond thousands of users, with keystroke dynamics constantly collected during the users' daily work. In this work, an active authentication learning and prediction system is developed based on a free-text keystroke dataset collected from approximately 2000 participants, which is much larger than the datasets reported in the works by Gunetti and Picardi (with 205 participants) and Villani et. al. (with 118 participants). To the best of our knowledge, the free-text keystroke dataset studied in this paper is the largest in literature.

Some researchers may attempt to use the same keyboard throughout the data collecting process. As pointed out by Villani et. al.'s work [36], the identification accuracy is prone to keyboard selection. In real world applications, it may be unrealistic to assume the keystroke dynamics to be collected from keyboards with the same keyboard model. In this study, the keystroke dynamics are collected through browser app, where no assumptions are made on the keyboard from which the keystroke dynamics are collected.

III. System Overview

A. Cognitive Factors in Keystroke Dynamics

By measuring the time stamps at each key press and key release events, various features can be extracted from the keystroke dynamics such as the dwell time of a monograph (the time length of a key-press); the time interval between two consecutive keystrokes in a digraph; the time duration between the first and last keystrokes in a trigraph or n-graph, etc.

Conventional keystroke dynamics usually do not distinguish the timing difference between different words, but only consider a collection of digraph latencies. Fig.1(a) illustrates a collection of digraph latencies ("re") observed from the same user, but are collected from four different words: "really", "were", "parents", and "store". It shows that a user's typing behavior is not only dependent on digraphs, but also highly dependent on words. On the other hand, Fig.1(b) illustrates the typing pattern of two users on the same word "really". It shows that the keystroke pattern of a word as a whole is user dependent.

In the work by Chang et. al. [46] and Wu et. al. [47], instead of breaking words into digraphs whose statistics are analyzed individually, they consider the correlation information between multiple keystroke intervals within a word, that is not revealed by digraph features. However, one serious concern is the lack of samples for each word, as the massive amount of English vocabulary dilutes the number of samples available for one particular word. Except for several frequently-used vocabulary such as "and", "are", "the", "to", etc, the lack of samples renders any pattern recognition technique to yield statistically sound decision rules. In order to preserve the correlation information between keystroke intervals within a word, while still retain sufficient amount of training samples, in this study we consider the correlation between the three consecutive keystroke time intervals in each trigraph.

More elaborately, in contrast to previous literature [6] which usually considered the total time duration of a trigraph, in our study a trigraph is represented by a three-dimensional vector, where each element in the vector is a time interval between two consecutive keystrokes. For instance, the word "really" which contains six consecutive time intervals

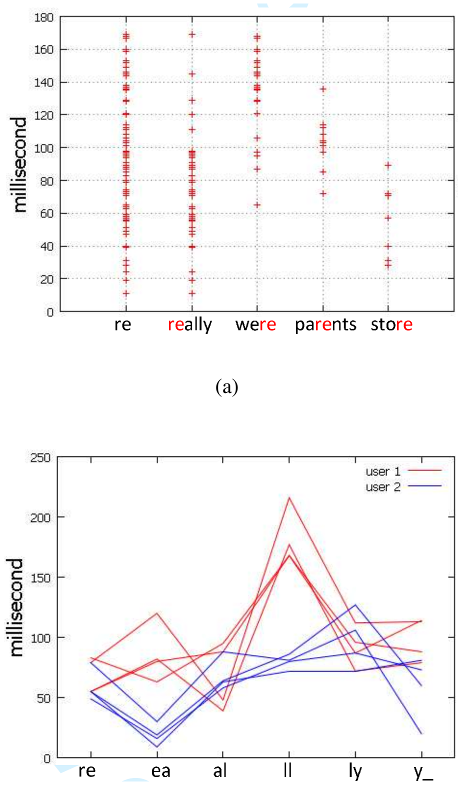
$$\mathrm{r} \overset{t_1}{-} \mathrm{e} \overset{t_2}{-} \mathrm{a} \overset{t_3}{-} \mathrm{l} \overset{t_4}{-} \mathrm{l} \overset{t_5}{-} \mathrm{y} \overset{t_6}{-} \text{(space)}$$

will be separated into four trigraphs each represented by a three dimensional vector, namely "rea"$(t_1 t_2 t_3)$, "eal"$(t_2 t_3 t_4)$, "all"$(t_3 t_4 t_5)$, and "lly"$(t_4 t_5 t_6)$.

B. System Architecture

The authentication system is user-specific, where for each legitimate user a profile is trained to recognize him as the only legal user. The authentication process only involves comparing the received sample to the user profile of the claimed identity, and is independent of other users' profiles in the system. The separated user profiles make it easier to update the system if the individual typing patterns change over time, and the entire system does not need to be retrained to add new users. Furthermore, the prediction time does not depend on the number of user profiles in the system.

As illustrated in Fig.2, the user profile of a legitimate user A contains a collection of most frequent trigraphs T_A, where each trigraph $w \in T_A$ accompanies a classifier h_{Aw} that evaluates user A's keystroke typing pattern of trigraph w. In continuous authentication process where an user B claims the identity of user A and types a word of $M \geq 3$ characters $c_1 c_2 \cdots c_M$, a total of $M-2$ trigraphs $w_i = c_i c_{i+1} c_{i+2}$, $i = 1, \cdots, M-2$ will be collected. If a trigraph w_i is one of the most frequent trigraphs by user A, namely $w_i \in T_A$, the trigraph classifier h_{Aw_i} will give a vote on whether or not user B should be authenticated as user A. The votes from all the trigraphs in T_A that user B types are then collected and weighted summed to arrive at the final decision. The details for determining the weights are discussed in Sec.VI.



(a)

(b)

Fig. 1. (a) Digraph "re" from the same user in different words. (b) Two users typing the same word "really".

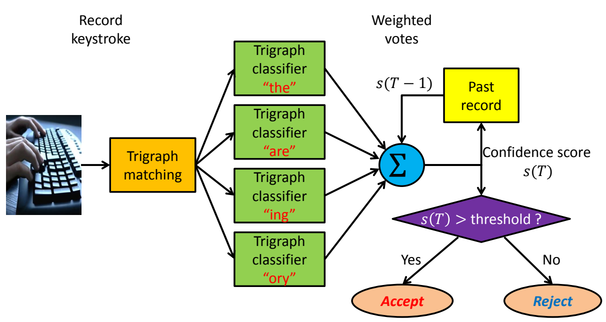


Fig. 2. The authentication system architecture.

IV. Learning Model Formulation

To train the decision boundary of a trigraph classifier h_{Aw} which summarizes user A's typing behavior on trigraph w, we formulate a binary classification problem by partitioning all training samples of trigraph w into two classes. The positive (legitimate) class comprises of samples collected from user A, while the negative (impostor) class is composed of samples from all users other than A.

Suppose there are N samples of trigraph w available for training, the training data set can be represented as $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$, where $\mathbf{x}_i \in \mathbb{R}^3$ is the feature vector and $y_i \in \{\pm 1\}$ is the label, indicating the sample either belongs to the positive class ($y_i = +1$) or negative class ($y_i = -1$).

A. Kernel Methods

The basic insight behind kernel trick is to nonlinearly transform patterns into some high-dimensional feature space, where various linear pattern recognition methods apply. The high dimensional feature space as well as the nonlinear mapping is determined by a kernel function that describes the similarity between pairwise samples, which should satisfy Mercer condition [24]. By Mercer's Theorem [24], a kernel function that satisfies Mercer's condition can be represented as the inner product in a kernel-induced feature space $\mathcal{H}$, namely $k(\mathbf{x}, \mathbf{x}') = \langle \boldsymbol{\phi}(\mathbf{x}), \boldsymbol{\phi}(\mathbf{x}') \rangle_{\mathcal{H}}$, where $\boldsymbol{\phi}(\mathbf{x})$ is some fixed mapping to $\mathcal{H}$. Common examples include the Gaussian RBF kernel

$$k_{RBF}(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2\sigma^2}\right) \tag{1}$$

and the polynomial kernel

$$k_{Poly\_p}(\mathbf{x}, \mathbf{x}') = \left(1 + \frac{\mathbf{x}^T \mathbf{x}'}{\sigma^2}\right)^p. \tag{2}$$

B. Kernel Ridge Regression

Denote kernel-based regression function

$$h(\mathbf{x}) = \langle \mathbf{u}, \boldsymbol{\phi}(\mathbf{x}) \rangle_{\mathcal{H}} \tag{3}$$

The design objective for kernel ridge regression [48]–[51] is to find a decision vector $\mathbf{u} \in \mathcal{H}$ that minimizes the regularized empirical risk [26]:

$$\min_{\mathbf{u} \in \mathcal{H}} \sum_{i=1}^N L(h(\mathbf{x}_i), y_i) + \rho \|\mathbf{u}\|_{\mathcal{H}}^2 \tag{4}$$

In dual variables [52], the regularized empirical risk (cf. (4)) can be rewritten as

$$\min_{\mathbf{a} \in \mathbb{R}^N} \sum_{i=1}^N L(h(\mathbf{x}_i), y_i) + \rho \mathbf{a}^T \mathbf{K} \mathbf{a} \tag{5}$$

where $[\mathbf{K}]_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$ is the kernel matrix, $\mathbf{a} = [a_1 \ \cdots \ a_N]^T$, and

$$h(\mathbf{x}) = \sum_{i=1}^N a_i k(\mathbf{x}_i, \mathbf{x}) \tag{6}$$

C. Class Dependent Costs for Imbalanced data set

Consider the weighted squared error empirical risk in the following form

$$L(h(\mathbf{x}), y) = c(y)(h(\mathbf{x}) - y)^2 \tag{7}$$

where $c(y) \in \mathbb{R}^+$ is a class-dependent weight. The regularized empirical risk becomes

$$\begin{aligned} &\min_{\mathbf{a} \in \mathbb{R}^N} \sum_{i=1}^N c(y_i) \left(\sum_{j=1}^N a_j k(\mathbf{x}_j, \mathbf{x}_i) - y_i \right)^2 + \rho \mathbf{a}^T \mathbf{K} \mathbf{a} \\ &= \min_{\mathbf{a} \in \mathbb{R}^N} \|\mathbf{K}\mathbf{a} - \mathbf{y}\|_{\mathbf{C}}^2 + \rho \mathbf{a}^T \mathbf{K} \mathbf{a} \end{aligned} \tag{8}$$

where $\|\mathbf{r}\|_{\mathbf{C}}^2 = \mathbf{r}^T \mathbf{C} \mathbf{r}$ is the Mahalanobis norm, $\mathbf{C}$ is a diagonal matrix with $C_{ii} = c(y_i)$, and $\mathbf{y} = [y_1 \ \cdots \ y_N]^T$.

Since (8) is convex and differentiable, it can be minimized by setting its derivative w.r.t. $\mathbf{a}$ equal to zero, giving the optimal solution

$$\mathbf{a} = (\mathbf{K} + \rho\mathbf{C}^{-1})^{-1}\mathbf{y} \tag{9}$$

Since the positive class contains only the legitimate user while the negative class contains all other users as impostors, the binary training data set is highly imbalanced in nature, where the positive class is outnumbered by the negative class. To avoid tendency for classifiers originally designed for balanced data sets to overlook the minorities and give poor results, we impose class-dependent costs and assign higher costs for misclassifying a positively-labeled sample. The class-dependent costs could be also based on the false-positive and false-negative costs, or on the prior probability of an impostor in practice for a more decision-theoretic approach. In this manuscript, the costs for misclassifying positive/negative samples are set to be inversely proportional to their population. More precisely, let N_+, N_- be the number of samples in positive/negative classes, respectively, we take

$$c(+1) = \frac{N}{2N_+}, \quad c(-1) = \frac{N}{2N_-} \tag{10}$$

D. Class Dependent Costs for SVM

To impose class dependent costs on SVM, we consider weighted hinge loss as empirical risk

$$L(h(\mathbf{x}), y_i) = c(y)\left[1 - y(h(\mathbf{x}) - y)\right]_+$$

The regularized empirical risk function (cf. (4)) becomes

$$\begin{array}{ll} \text{minimize} & \frac{\rho}{2}\|\mathbf{u}\|^2_{\mathcal{H}} + \sum_{i=1}^N c(y_i)\xi_i \\ \text{subject to} & y_i(\langle \mathbf{u}, \phi(\mathbf{x}_i)\rangle_{\mathcal{H}} + b) \geq 1 - \xi_i \\ \text{variables} & \mathbf{u} \in \mathcal{H}, b \in \mathbb{R}, \xi_i \geq 0, i = 1, ..., N \end{array} \tag{11}$$

which can be solved by LIB-SVM [53] with class-dependent cost parameters $\frac{c(y_i)}{\rho}$, more explicitly,

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}\boldsymbol{\alpha}^T\mathbf{K}\boldsymbol{\alpha} - \mathbf{e}^T\boldsymbol{\alpha} \\ \text{subject to} & \mathbf{y}^T\boldsymbol{\alpha} = 0 \\ \text{variables} & 0 \leq \alpha_i \leq \frac{c(y_i)}{\rho}, i = 1, ..., N. \end{array} \tag{12}$$

V. Improving Classification Complexity of Kernel-Based Classifiers

Based on our previous work [27] on cost-efficient KRR algorithms, our system enables trade-off between classification/learning complexity and accuracy performance by means of selecting appropriate finite decomposable kernel function.

A. Decision Function in Kernel Induced Feature Space

For finite decomposable kernel function, whose kernel-induced feature space $\mathcal{H} \subseteq \mathbb{R}^J$ has finite dimensions and Euclidean inner product

$$k(\mathbf{x}, \mathbf{x}') = \sum_{j=1}^J \phi^{(j)}(\mathbf{x})\phi^{(j)}(\mathbf{x}') = \boldsymbol{\phi}(\mathbf{x})^T\boldsymbol{\phi}(\mathbf{x}') \tag{13}$$

The regression function can be rewritten as

$$h(\mathbf{x}) = \sum_{i=1}^N a_i\boldsymbol{\phi}(\mathbf{x}_i)^T\boldsymbol{\phi}(\mathbf{x}) = \mathbf{u}^T\boldsymbol{\phi}(\mathbf{x}) \tag{14}$$

where the decision vector $\mathbf{u} = \sum_{i=1}^N a_i\boldsymbol{\phi}(\mathbf{x}_i)$ can be pre-computed in the learning phase.

Given a test pattern $\mathbf{x}$, it requires $O(J)$ operations to produce all elements of $\boldsymbol{\phi}(\mathbf{x})$, and another $O(J)$ operations to compute the inner product $\mathbf{u}^T\boldsymbol{\phi}(\mathbf{x})$. Therefore the total classification complexity is $O(J)$, which is independent of N.

In this paper, one important kernel in consideration is the p-th order polynomial kernel (cf. (2)), abbreviated as POLY_p, whose basis functions take the following form

$$\phi^{(j)}(\mathbf{x}) = \sqrt{\frac{p!}{(p-\ell)!}} \prod_{m=1}^M \frac{1}{\sqrt{d_m!}}\left(\frac{x^{(m)}}{\sigma}\right)^{d_m} \tag{15}$$
$$0 \leq \ell \leq p, \ell = d_1 + \cdots + d_M$$

There are $J = J^{(p)} = \frac{(M+p)!}{M!p!}$ different combinations.

The flexibility in classification schemes results in a classification complexity of $O(\min(NM, J))$. More elaborately, for small datasets with less number of training samples N, (6) is adopted with a classification cost of $O(NM)$. On the contrary, for large datasets, one may adopt (14) instead of (6) to achieve a $O(J)$ classification cost, which is constant and independent of the size of the training dataset.

B. Finite p-Degree Approximation of RBF Kernel

The TRBF kernel [27] is defined as

$$\begin{aligned} k_{TRBF}(\mathbf{x}, \mathbf{x}') &= \exp\left(-\frac{\|\mathbf{x}\|^2}{2\sigma^2}\right)\left(\sum_{\ell=1}^p \frac{1}{\ell!}\left(\frac{\mathbf{x}^T\mathbf{x}'}{\sigma^2}\right)^\ell\right)\exp\left(-\frac{\|\mathbf{x}'\|^2}{2\sigma^2}\right) \\ &= \boldsymbol{\phi}(\mathbf{x})^T\boldsymbol{\phi}(\mathbf{x}') \end{aligned} \tag{16}$$

where each basis function takes the following form

$$\phi^{(j)}(\mathbf{x}) = \exp\left(-\frac{\|\mathbf{x}\|^2}{2\sigma^2}\right)\prod_{m=1}^M \frac{1}{\sqrt{d_m!}}\left(\frac{x^{(m)}}{\sigma}\right)^{d_m} \tag{17}$$
$$0 \leq d_1 + \cdots + d_M \leq p$$

The trade-off between accuracy performance and computation efficiency highly depends on order p and its intrinsic dimension $J = J^{(p)}$, which is identical to that of polynomial kernels. In this paper we refer to TRBF kernels with order p as TRBF_p. Note that TRBF is simply a Taylor expansion approximation of RBF. For a more sophisticated RBF approximation, see [54].

C. Comparison Between POLY and TRBF Kernels

Despite the similar appearance between POLY and TRBF kernel (cf. (15),(17)), they have the following distinctions:

- POLY_p has an additional multiplication factor $\sqrt{\frac{p!}{(p-\ell)!}}$, which increases with the monomial order ℓ and hence amplifies the high-order terms. That is to say, TRBF kernel imposes less weights on high order terms than polynomial kernels.
- TRBF_p has an additional multiplication factor $\exp\left(-\frac{\|\mathbf{x}\|^2}{2\sigma^2}\right)$, which forces its basis functions (cf. (17)) to converge to zero as the magnitude of $\mathbf{x}$ grows to infinity, making it more suitable for forming closed, local decision boundaries. On the contrary, the basis functions

deduced by POLY_p (cf. (15)) will grow unbounded as $\|\mathbf{x}\|$ grows to infinity, making it more sensitive to outliers.

- TRBF_p converges to the commonly adopted RBF kernel as degree p increases towards infinity. On the contrary, POLY_p diverges as degree p increases towards infinity.

We refer to Kung's book [23] for more details on the properties of TRBF kernel.

D. Fast Learning Kernel Methods

For finite decomposable kernel function (cf. (13)), the kernel matrix is tightly linked to the training inputs in $\mathcal{H}$:

$$\mathbf{K} = \mathbf{\Phi}^T \mathbf{\Phi} \tag{18}$$

where $\mathbf{\Phi} = [\boldsymbol{\phi}(\mathbf{x}_1) \ \cdots \ \boldsymbol{\phi}(\mathbf{x}_N)]$ is the data matrix in kernel-induced feature space.

1) Learning Complexity of SVM: The SVM learning involves a quadratic programming problem with learning complexity at least $\Omega(N^2)$. For RBF kernel, which has infinite dimensional kernel induced feature space, the number of support vectors usually increases with the number of training samples N, which tends to further increase its learning cost.

2) Learning Complexity for KRR: The KRR learning focuses on solving the decision vector $\mathbf{a}$ in (9), which involves inverting a $N \times N$ matrix $(\mathbf{K} + \rho \mathbf{C}^{-1})$ and therefore demands a high complexity of $O(N^3)$.

The quadratic and cubic growth with the number of training samples N renders SVM and KRR from being computationally affordable in large scale learning problems. In numbers, in our experiment there are approximately $N \approx 80000$ samples for the popular word "the" in the dataset, resulting in learning cost of the order $80000^3 \approx 10^{15}$, which is impractical and calls for a cost-efficient KRR algorithm. Several methods were proposed to relieve computation burden [49], [55], [56]. In this work we implement a cost-efficient algorithm [27] whose learning complexity grows linearly with N in the active authentication problem, as described as below.

3) Fast Algorithm for KRR: Let us rewrite the regularized weighted squared error empirical risk as

$$\sum_{i=1}^{N} c(y_i)(h(\mathbf{x}_i) - y_i)^2 + \rho \|\mathbf{u}\|^2_{\mathcal{H}} = \|\mathbf{\Phi}^T \mathbf{u} - \mathbf{y}\|^2_{\mathbf{C}} + \rho \|\mathbf{u}\|^2 \tag{19}$$

and set its partial derivatives to zero, we may solve the decision vector in explicit form

$$\mathbf{u} = (\mathbf{\Phi} \mathbf{C} \mathbf{\Phi}^T + \rho \mathbf{I})^{-1} \mathbf{\Phi} \mathbf{C} \mathbf{y} \tag{20}$$

The fast-KRR algorithm solves decision vector $\mathbf{u}$ instead of $\mathbf{a}$, which incurs three costs: (1) The computation of the $J \times J$ matrix $\mathbf{\Phi} \mathbf{C} \mathbf{\Phi}^T$ requires $O(J^2 N)$ operations; (2) The inversion of $(\mathbf{\Phi} \mathbf{C} \mathbf{\Phi}^T + \rho \mathbf{I})$ requires $O(J^3)$ operations; (3) The matrix-vector multiplication requires a negligible $O(NJ)$ operations. In summary, the learning complexity is $O(J^3 + J^2 N)$, which is linear w.r.t. N.

VI. Fusion Methods

In Chair et al.'s work [57], a fusion scheme is proposed which combines decisions from multiple independent classifiers by weighted votes. The weights depend not only on the classifier, but also on its outcome. The baseline is that information provided by acceptance or rejection is not equal and is dependent on the classifier's false rejection rate (FRR) and false acceptance rate (FAR). Intuitively speaking, for a classifier with very low FRR but rather moderate FAR, since false rejection is more unlikely than false acceptance, its rejection votes would have larger weights compared to acceptance votes. On the other hand, for a classifier with moderate FRR but very low FAR, its acceptance votes should be more persuasive than rejection votes.

Following their concepts, in this study there are two weights accompanying with each word classifier h_{Aw}, namely the acceptance weight $\beta_{Aw}^{(acc)}$ and the rejection weight $\beta_{Aw}^{(rej)}$. Both weights are determined by the estimated FAR (denoted as $\hat{p}_{FAR}$) and FRR (denoted as $\hat{p}_{FRR}$) as follows

$$\beta_{Aw}^{(acc)} = \log\left(\frac{1-\hat{p}_{FRR}}{\hat{p}_{FAR}}\right), \quad \beta_{Aw}^{(rej)} = \log\left(\frac{1-\hat{p}_{FAR}}{\hat{p}_{FRR}}\right) \tag{21}$$

The authentication process maintains a confidence score $s_{BA}(T)$ representing how confident the system is to authenticate user B as user A at time stamp T. If user B types a word which contains trigraph w at time stamp T, the confidence score is updated as

$$s_{BA}(T) = \begin{cases} s_{BA}(T-1) + \beta_{Aw}^{(acc)} & \text{(accept)} \\ s_{BA}(T-1) - \beta_{Aw}^{(rej)} & \text{(reject)} \end{cases} \tag{22}$$

There is a Bayesian interpretation of (21) [57]. Let $p_{legi}^{(pre)}$, $p_{hack}^{(pre)}$ be the prior probabilities of user B being the legitimate user A or impostor, respectively. By Bayes rule, if word classifier h_{Aw} gives an acceptance vote, the posterior probabilities $p_{legi}^{(post)}$, $p_{hack}^{(post)}$ are given by

$$p_{legi}^{(post)} = \frac{p_{legi}^{(pre)}(1-\hat{p}_{FRR})}{p_{legi}^{(pre)}(1-\hat{p}_{FRR}) + p_{hack}^{(pre)}\hat{p}_{FAR}} \tag{23a}$$

$$p_{hack}^{(post)} = \frac{p_{hack}^{(pre)}\hat{p}_{FAR}}{p_{legi}^{(pre)}(1-\hat{p}_{FRR}) + p_{hack}^{(pre)}\hat{p}_{FAR}} \tag{23b}$$

The logarithm of the ratio between p_{legi} and p_{hack} is therefore updated as

$$\log\left(\frac{p_{legi}^{(post)}}{p_{hack}^{(post)}}\right) = \log\left(\frac{p_{legi}^{(pre)}}{p_{hack}^{(pre)}} \frac{1-\hat{p}_{FRR}}{\hat{p}_{FAR}}\right) = \log\left(\frac{p_{legi}^{(pre)}}{p_{hack}^{(pre)}}\right) + \beta_{Aw}^{(acc)}$$

Similarly, if the word classifier gives a rejection vote, the posterior probabilities are given by

$$p_{legi}^{(post)} = \frac{p_{legi}^{(pre)}\hat{p}_{FRR}}{p_{legi}^{(pre)}\hat{p}_{FRR} + p_{hack}^{(pre)}(1-\hat{p}_{FAR})} \tag{24a}$$

$$p_{hack}^{(post)} = \frac{p_{hack}^{(pre)}(1-\hat{p}_{FAR})}{p_{legi}^{(pre)}\hat{p}_{FRR} + p_{hack}^{(pre)}(1-\hat{p}_{FAR})} \tag{24b}$$

Analogously, one has

$$\log\left(\frac{p_{legi}^{(post)}}{p_{hack}^{(post)}}\right) = \log\left(\frac{p_{legi}^{(pre)}}{p_{hack}^{(pre)}}\right) - \beta_{Aw}^{(rej)}$$

Compare with (22), the confidence score can be mathematically interpreted as

$$s_{BA}(T) = \log\left(\frac{p_{legi}(T)}{p_{hack}(T)}\right) \tag{25}$$

where $p_{legi}(T)$, $p_{hack}(T)$ denotes the system's belief about the user being legitimate or imposter at time T.

In this study, the FAR and FRR performances are estimated by 3-fold cross validation with Bayesian average:

$$\hat{p}_{FRR} = \frac{\#\text{false rejection} + 1}{\#\text{rejection} + 2}, \quad \hat{p}_{FAR} = \frac{\#\text{false acceptance} + 1}{\#\text{acceptance} + 2}$$

VII. Experiment

A. Experiment Assembly

To verify the cost-performance trade-off, we conduct experiments on free-text keystroke dataset collected by Chang et al. [46]. The dataset contains keystroke dynamics collected by web-based software system from 1977 students in Iowa State University. The system provided three segments (Segment I, II, III) of simulated user environments, including typing short sentences, writing short essays, and browsing web-pages. Each segment takes approximately 30 minutes to be completed by a participant. In this study we only analyze the twenty-six lower-case letters plus the space, where we regard the upper-case letters as their lower-case letter counterparts.

Among all 1977 participants, there were 18 participants whose data were manually discarded due to one or multiple of the following reasons:

- They quit in the middle of the experiment.
- They repeatedly typed in meaningless words, such as "fdsafewaqfsdagsa fd df d fsd af dsa fs a f af f f ff f".
- They used touch screen instead of keyboard to conduct the experiment.

Among the remaining 1959 participants, there were 978 participants who completed all the three segments I, II, III, while the other 981 participants completed only segments I, II. In the following text, we denote set U as the 978 participants who completed all the three segments, and set U^c as the other 981 users who only completed segments I, II. Note that participants in U and U^c are disjoint.

During the training phase, the training dataset consists of keystroke dynamics collected in segments I, III from all participants in U, where each participant (also referred to as legitimate user) has approximately 2100 words collected. Each legitimate user $A \in U$ has his own profile trained by formulating a binary classification problem, where the positive class consists of keystroke dynamics collected from A himself, and the negative class consists of keystroke dynamics collected from a random subset of 100 users in $U - A$.

During the testing phase, the test dataset consists of keystroke dynamics collected in segment II from all participants in either U^c (also referred to as impostors) or U. There are approximately 900 words collected from each participant as test data.

B. Parameter Selection

To select kernel bandwidth σ (cf. (1)) and regularization parameter ρ (cf. (12)) for SVM-RBF, we perform 3-fold cross validation on training dataset as to be elaborated as below: For each legitimate user $A \in U$, we take keystroke dynamics from user A in training dataset (segments I, III) as positive class, and keystroke dynamics from a random subset of 50 users in $U - A$ in training dataset as negative class. The occurrences of false rejection and false acceptance for authenticating $A \in U$ are then evaluated by 3-fold cross validation. Table. I, II summarizes the evaluated EER and the area under detection error rate curve (AUC) on training dataset for $\sigma = 0.1, 0.2, 0.5, 1, 3$ and $\rho = 0.5, 1, 2, 5$ (cf. (12)). We choose $\sigma = 0.5$, $\rho = 2$, which minimizes both EER and AUC evaluated by cross-validation on training dataset. For KRR-TRBF and KRR-POLY, we choose $\sigma = 0.5$ and select the corresponding ρ which minimizes the EER evaluated by cross validation on training dataset, as summarized in Table.III. The confidence score threshold at which the EER in Table III is achieved is also summarized in Table IV.

C. Performance Metrics

The main performance metrics include the FRR and FAR, which are measured as follows:

- *FRR:* A false rejection is detected whenever a profile of a legitimate user $A \in U$ fails to accept himself as the legitimate user. The authentication system (cf. Figure 2) will compare the keystroke dynamics of every word user A typed in the testing phase (a.k.a. segment II) with his own profile to see if the final confidence score, which is a weighted sum of votes from the various trigraph classifiers in profile, is beyond the threshold. The reported FRR is defined as
$$FRR = \frac{1}{|U|}\sum_{i\in U} 1\{\text{Profile i rejects user i}\}.$$
Here $1\{\cdot\}$ denotes the indicator function.
- *FAR:* Each of the 978 profiles for legitimate users in U is attacked by all the 981 impostors in U^c. For an imposter $B \in U^c$ to claim the identity of user $A \in U$, the authentication system (cf. Figure 2) compares the keystroke dynamics of every word imposter B typed in the testing phase with A's profile to see if the final confidence score is beyond the threshold. A false acceptance is detected whenever a legitimate user profile accepts an impostor as the legitimate user. More precisely, the reported FAR is defined as
$$FAR = \frac{1}{|U||U^c|}\sum_{i\in U}\sum_{j\in U^c} 1\{\text{Profile i accepts user j}\}.$$

The detection error trade-off (DET) curves in Figures 3,4,5 are plotted by tuning the confidence score threshold in Figure 2 to trade-off between FRR and FAR. The EER and AUC, as well as the confidence score at which EER is obtained, are summarized in Table.V.

TABLE I
EER OF SVM-RBF EVALUATED BY 3-FOLD CROSS VALIDATION FROM TRAINING DATASET.

EER	$\rho = 0.5$	$\rho = 1$	$\rho = 2$	$\rho = 5$
$\sigma = 0.1$	29.79%	27.47%	24.02%	17.88%
$\sigma = 0.2$	9.22%	7.11%	4.02%	2.93%
$\sigma = 0.5$	1.18%	0.89%	0.76%	0.78%
$\sigma = 1$	0.92%	0.79%	0.97%	1.53%
$\sigma = 3$	1.39%	2.46%	5.01%	16.80%

TABLE II
AUC OF SVM-RBF EVALUATED BY 3-FOLD CROSS VALIDATION FROM TRAINING DATASET.

AUC	$\rho = 0.5$	$\rho = 1$	$\rho = 2$	$\rho = 5$
$\sigma = 0.1$	2.08×10^{-1}	1.65×10^{-1}	1.25×10^{-1}	8.38×10^{-2}
$\sigma = 0.2$	2.68×10^{-2}	1.45×10^{-2}	5.63×10^{-3}	2.85×10^{-3}
$\sigma = 0.5$	6.56×10^{-4}	4.80×10^{-4}	4.60×10^{-4}	5.30×10^{-4}
$\sigma = 1$	5.36×10^{-4}	6.10×10^{-4}	7.78×10^{-4}	1.50×10^{-3}
$\sigma = 3$	1.47×10^{-3}	3.52×10^{-3}	1.02×10^{-2}	6.16×10^{-2}

TABLE III
EER OF KRR-TRBF AND KRR-POLY EVALUATED BY 3-FOLD CROSS VALIDATION FROM TRAINING DATASET.

EER	linear	TRBF2	TRBF3	POLY2	POLY3
$\rho = 0.01$	0.93%	1.16%	1.98%	1.00%	2.34%
$\rho = 0.05$	1.00%	0.98%	1.60%	1.13%	1.97%
$\rho = 0.1$	0.93%	1.09%	1.50%	0.98%	1.57%
$\rho = 0.5$	0.92%	0.95%	1.18%	1.00%	1.47%
$\rho = 1$	1.02%	0.76%	0.87%	1.02%	1.33%
$\rho = 5$	1.19%	0.78%	0.95%	0.77%	1.09%
$\rho = 10$	1.22%	0.89%	0.89%	0.74%	0.93%

TABLE IV
CONFIDENCE SCORE THRESHOLD FOR KRR-TRBF AND KRR-POLY AT WHICH THE REPORTED EER IN TABLE III IS ACHIEVED.

EER	linear	TRBF2	TRBF3	POLY2	POLY3
$\rho = 0.01$	-2	-12	-15	-12	-14
$\rho = 0.05$	-3	-12	-15	-13	-14
$\rho = 0.1$	-3	-12	-15	-12	-13
$\rho = 0.5$	-3	-13	-14	-13	-15
$\rho = 1$	-3	-11	-12	-14	-14
$\rho = 5$	-2	-11	-14	-11	-14
$\rho = 10$	-2	-11	-13	-10	-13

TABLE V
EER AND AUC UNDER DET CURVE COMPARISON.

Kernel	EER	AUC	Conf. Thresh.
KRR-linear	1.80%	0.00249	-13
KRR-POLY2	1.53%	0.00164	-26
KRR-POLY3	1.43%	0.00189	-24
SVM-RBF	1.41%	0.00203	-31
KRR-TRBF2	1.74%	0.00162	-27
KRR-TRBF3	1.39%	0.00182	-25

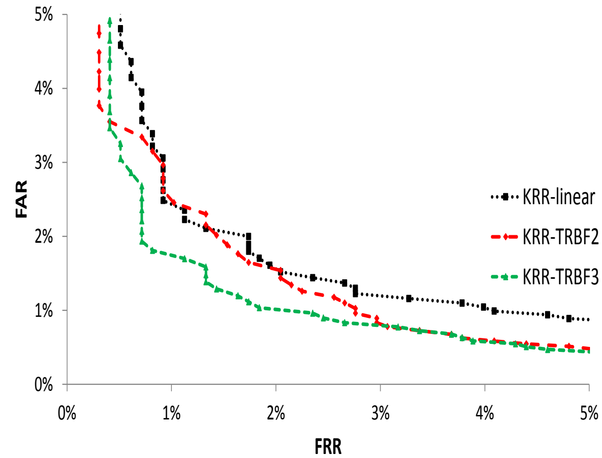


Fig. 3. DET curves for KRR learning model with TRBF kernel of various degrees.

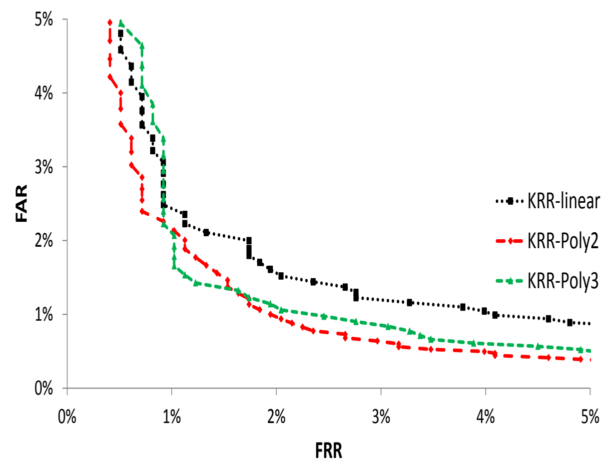


Fig. 4. DET curves for KRR learning model with polynomial kernel of various degrees.

*1) **Error Rates for KRR-TRBF and KRR-POLY with Various Degrees:*** Figure 34 summarizes the detection error trade-off (DET) curves for KRR learning model with TRBF and POLY kernels of various degrees. In terms of equal error rates, we observe that

$$KRR - TRBF3 < KRR - TRBF2 < KRR - linear.$$

$$KRR - POLY3 < KRR - POLY2 < KRR - linear.$$

The EER for both KRR-TRBFp and KRR-POLYp decreases as their degree p increases. This can be explained by the higher dimension J of its kernel-induced feature space $\mathcal{H} = \mathbb{R}^J$, which provides stronger representation power.

*2) **Comparison between KRR and SVM-RBF:*** Figure 5 shows the DET curves for KRR learning model with TRBF3 and Poly3 kernels, namely KRR-TRBF3 and KRR-Poly3, respectively. They are compared to the SVM learning model with Gaussian RBF kernel as a benchmark. We observe that KRR-TRBF3, KRR-Poly3, and SVM-RBF have very similar EER. However, KRR-TRBF3 has signicantly lower FAR concerning the region where FRR is less than 1%. In terms of AUC (under

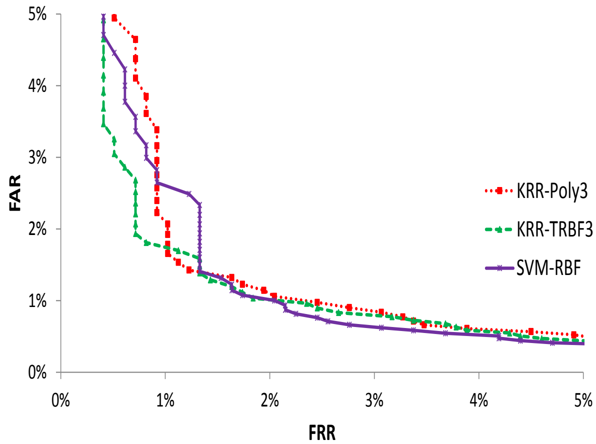


Fig. 5. DET curves for KRR-TRBF3, KRR-Poly3, and SVM-RBF.

DET curve), KRR-TRBF3 outperforms both SVM-RBF and KRR-POLY3.

D. Scalability Issues

Besides error rates, it is also an important issue on how the training and prediction computational costs of a learning model scales with the size of the collected data. The training time and prediction time reported in Figures 6,7,8,9 are measured as follows

- *Training time:* Let $t_{train}^{(i)}$ be the time needed to train the profile for legitimate user i. We report the averaged training time defined as

$$t_{train\_avg} = \frac{1}{|U|}\sum_{i\in U} t_{train}^{(i)},$$

- *Prediction time:* Let $t_{pred}^{(ij)}$ be the prediction time for comparing the typing patterns by imposter j to the profile of legitimate user i. We report the averaged prediction time defined as

$$t_{pred\_avg} = \frac{1}{|U||U^c|}\sum_{i\in U}\sum_{j\in U^c} t_{pred}^{(ij)},$$

The simulations are conducted on two Intel Xeon X5680 CPU @3.33 GHz, 8 GB RAM, with 6 cores for each processor, running the Linux version 2.6.32 with Red Hat 4.4.7-4 version.

To see how training time scales up with the training data size, we conduct experiment to be elaborated as follows: In the training phase, the profile for a legitimate user $A \in U$ is trained by formulating a binary classification problem. Similar to the experimental setup in Sec.VII-A, the positive class is composed of keystroke dynamics collected from user A in segment I, III. The negative class, however, is composed of keystroke dynamics collected from a random subset of L users in $U - A$, where L is a tunable integer which is roughly proportional to the training data size. In the following experiments we take $L \in \{50, 100, 150, 200, 250, 300\}$.

Since the TRBF_p and Poly_p kernels have exactly the same kernel-induced Hilbert space dimension $J^{(p)}$, they have almost identical training and prediction costs, which is also observed

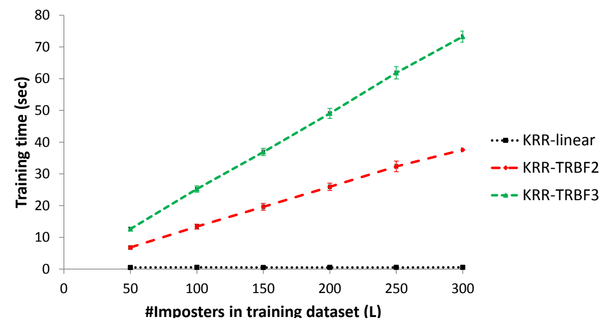


Fig. 6. Training time for KRR learning model with TRBF kernel of various degrees.

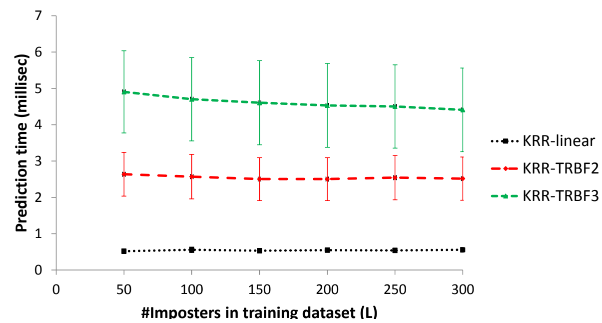


Fig. 7. Prediction time for KRR learning model with TRBF kernel of various degrees.

by our experiments. In the following context, we will focus on the training and prediction costs for TRBF kernels.

1) **Training time for KRR-TRBF with various degrees:** Figure 6 summarizes the training time for KRR learning model with TRBF kernel of various degrees. We observe that for each specific curve, the training time grows linearly with L, which is roughly proportional to the training data size as expected. Recall Figure 3, we also observe a consistent trade-off between error rate performance and training time: With higher degree p, the TRBF_p kernel has higher kernel-induced Hilbert space dimension $J^{(p)}$, which implies stronger representation power and smaller error rates, at a cost of higher training cost.

2) **Prediction time for KRR-TRBF with various degrees:** Figure 7 summarizes the prediction time for KRR learning model with TRBF kernel of various degrees. We observe that for each specific curve, the prediction time is independent of L. In other words, the prediction time is constant over training data size. Recall Figure 3, we also observe a consistent trade-off between error rate performance and prediction time, where TRBF kernel with higher degree gives smaller error rates but requires higher prediction time.

3) **Training and Prediction Time Comparison between KRR and SVM:** Figure 8 plots the training time for KRR learning model with TRBF3 and Poly3 kernels. They are compared to the SVM learning model with Gaussian RBF kernel as a benchmark. We observe that both KRR-Poly3 and KRR-TRBF3 have significantly less training cost than SVM-RBF. Furthermore, the training time for both KRR-TRBF3 and KRR-Poly3 grow linearly with the training data size N, while

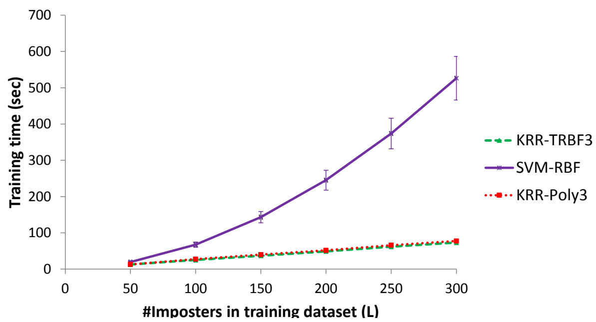


Fig. 8. Training time for KRR learning model with TRBF3 and Poly3 kernels, which are compared with SVM learning model with Gaussian-RBF kernel.

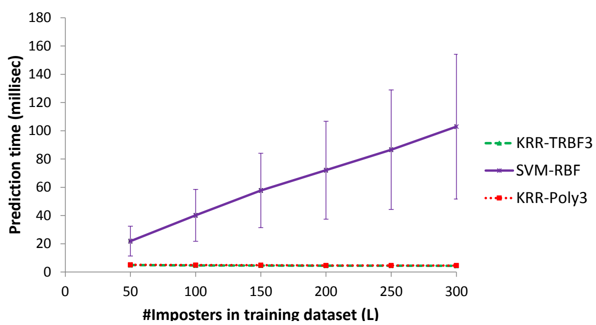


Fig. 9. Prediction time for KRR learning model with TRBF3 and Poly3 kernels, which are compared with SVM learning model with Gaussian-RBF kernel.

SVM-RBF has training time growing quadratically with N.

Figure 9 plots the prediction time for KRR-TRBF3, KRR-Poly3, and SVM-RBF. We observe that both KRR-TRBF3, KRR-Poly3 have significantly less prediction cost than SVM-RBF. Furthermore, the prediction time for both KRR-TRBF3 and KRR-Poly3 remains constant regardless of training data size, while SVM-RBF has prediction time that scales up linearly with the training data size N.

Recall Figure 5, both KRR-POLY3 and KRR-TRBF3 achieve significantly less training and prediction times while retaining comparable error rates as SVM-RBF. This shows great potential in large-scale authentication system applications.

VIII. Discussions and Conclusions

In real world applications, an authentication system can easily grow beyond thousands of users, with keystroke dynamics constantly collected during the users' daily work. The large scale dataset raises scalability concerns, which in turn necessitate our development of efficient learning and prediction algorithms. We apply Kung and Wu's work [27] to (1) approximate the Gaussian-RBF kernel with a truncated-RBF (TRBF) kernel and (2) then solve the KRR learning model in the intrinsic space [27]. This results in a fast-KRR learning algorithm with $O(N)$ training cost, making it very cost effective for large-scale learning applications. Likewise, in the prediction phase, the RBF kernels again suffer from the curse of dimensionality problem, causing its prediction time to grow linearly with the training data size N, or more exactly, with the number of support vectors. In contrast, the TRBF kernel needs only a constant prediction time regardless of the training data size, rendering it very appealing for real-time prediction.

The fast-KRR algorithm (along with TRBF kernels) offers computational advantages over the traditional SVM with Gaussian-RBF kernel, while retaining similar error-rate performances. More precisely, our learning model achieves an equal error rate of 1.39% with $O(N)$ training time, while SVM with the RBF kernel shows a rate of 1.41% with $O(N^2)$ training time. This points to potentially promising deployment of the fast-KRR learning model for real-world large-scale active authentication systems. Furthermore, the TRBF kernel may be tuned by the TRBF order which in turn dictates the intrinsic degree J of the TRBF kernel. Both the theory and experiments shows that, by tuning the intrinsic degree J, one may strike a compromise between accuracy and training/prediction complexities.

Besides the class-dependent cost algorithmic approach implemented in this manuscript, there are various techniques proposed to ameliorate the class imbalance problem both on the algorithmic and data levels [58], [59]. At the data level, different forms of re-sampling are proposed such as random oversampling the minority class with replacement, random undersampling the majority class, directed oversampling, directed undersampling, oversampling with informed generation of new samples, or a combination of the aforementioned approaches [60]. At the algorithmic level, solutions include class-dependent costs to compensate class imbalance [61], adjusting the decision threshold, adopting recognition based (formulate as one-class problem) rather than discrimination-based (formulate as two class problem) learning. We will explore various data-centered approaches for class imbalanced problems in our future work.

In this manuscript the flexibility of hyper-parameter selection is not yet fully explored. For instance, the optimal hyper parameter σ for POLY and TRBF kernels may be different, as they weight higher order terms differently. Also, Table.III suggests that EER may be further reduced by selecting a wider range of hyper-parameter ρ. These issues will be further addressed in our future work.

The KRR-TRBF implemented in this manuscript can be considered as a regular linear regression in a finite dimensional space $\mathbb{R}^J$, where the raw attributes are mapped to $\mathbb{R}^J$ by some specific nonlinear transformation. Such idea of representing the samples by vectors in some finite dimensional space $\mathbb{R}^J$, on which the original kernel regression problem is approximated by a regular linear regression problem in $\mathbb{R}^J$, can be also found in other large-scale KRR approaches such as Nystrom method [62] and fixed-size LS-SVM [63]. The difference lies in how the finite dimensional space is formulated. In Nystrom method the principle component analysis (PCA) is implicitly applied on the N training samples in the kernel-induced feature space $\mathcal{H}$, where each sample is represented by its N principle components; In fixed size LSSVM [63], instead of performing PCA on all the N training samples, it selects a subsample of predefined size $J \ll N$ by maximizing the quadratic Renyi entropy, and then apply PCA on the selected J subsamples to find J principle components to represent each sample. In the future work we will quantitatively compare KRR-TRBF with Nystrom method and fixed-size LS-SVM, as well as other approaches summarized in [33] which are scalable for large-scale active authentication applications.

Acknowledgment

This work was supported in part by the Active Authentication Program of DARPA under grant FA8750-12-2-0200.[1]

References

[1] A. Adams and M. A. Sasse, "Users are not the enemy," *Commun. ACM*, vol. 42, no. 12, pp. 41–46, 1999.

[2] A. Peacock, X. Ke, and M. Wilkerson, "Typing patterns: A key to user identication," *IEEE Security Privacy*, vol. 2, no. 5, pp. 40–47, 2004.

[3] K. Niinuma, U. Park, and A. K. Jain, "Soft biometric traits for continuous user authentication," *IEEE Trans. Inf. Forensics Security*, vol. 5, no. 4, pp. 771–780, 12 2010.

[4] J. Daugman, "How iris recognition works," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 14, no. 1, p. 2004, 1 2004.

[5] T. Sim, S. Zhang, R. Janakiraman, and S. Kumar, "Continuous verification using multimodal biometrics," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 4, pp. 687–700, 4 2007.

[6] D. Gunetti and C. Picardi, "Keystroke analysis of free text," *ACM Trans. Inf. Syst. Secur*, vol. 8, no. 3, pp. 312–347, 8 2005.

[7] M. Pusara and C. E. Brodley, "User re-authentication via mouse movements," in *Proc. 2004 ACM Workshop Visualization and Data Mining for Computer Security*, ser. VizSEC/DMSEC '04. New York, NY, USA: ACM, 2004, pp. 1–8.

[8] J. Lu and Y. P. Tan, "Regularized locality preserving projections and its extensions for face recognition," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 40, no. 3, pp. 958–963, June 2010.

[9] M. F. Valstar, M. Mehu, B. Jiang, M. Pantic, and K. Scherer, "Meta-analysis of the first facial expression recognition challenge," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 42, no. 4, pp. 966–979, Aug 2012.

[10] J. Zuo and N. A. Schmid, "On a methodology for robust segmentation of nonideal iris images," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 40, no. 3, pp. 703–718, June 2010.

[11] Y. Du, E. Arslanturk, Z. Zhou, and C. Belcher, "Video-based noncooperative iris image segmentation," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 41, no. 1, pp. 64–74, Feb 2011.

[12] R. Cappelli, "Fast and accurate fingerprint indexing based on ridge orientation and frequency," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 41, no. 6, pp. 1511–1521, Dec 2011.

[13] R. Cappelli, M. Ferrara, and D. Maio, "A fast and accurate palmprint recognition system based on minutiae," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 42, no. 3, pp. 956–962, June 2012.

[14] M. Goffredo, I. Bouchrika, J. N. Carter, and M. S. Nixon, "Self-calibrating view-invariant gait biometrics," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 40, no. 4, pp. 997–1008, Aug 2010.

[15] M. Karg, K. Kuhnlenz, and M. Buss, "Recognition of affect based on gait patterns," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 40, no. 4, pp. 1050–1061, Aug 2010.

[16] R. Joyce and G. Gupta, "Identity authentication based on keystroke latencies," *Commun. ACM*, vol. 33, no. 2, pp. 168–176, 2 1990.

[17] R. Spillane, "Keyboard apparatus for personal identification," *IBM Tech. Disclosure Bulletin*, vol. 17, no. 3346, 1975.

[18] F. Girosi, M. Jones, and T. Poggio, "Regularization theory and neural networks architectures," *Neural Computation*, vol. 7, pp. 219–269, 1995.

[19] V. N. Vapnik, *The Nature of Statistical Learning Theory*. New York: Springer-Verlag, 1995.

[20] B. Scholkopf, A. Smola, and K.-R. Muller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Computation*, vol. 10, pp. 1299–1319, 1998.

[21] M. E. Mavroforakis and S. Theodoridis, "A geometric approach to support vector machine (svm) classification," *IEEE Trans. Neural Netw.*, vol. 17, no. 3, pp. 671–682, 5 2006.

[22] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, 2nd ed. New York: Wiley, 2001.

[23] S. Y. Kung, *Kernel Methods and Machine Learning*. Cambridge University Press, 2014.

[24] J. Mercer, "Functions of positive and negative type, and their connection with the theory of integral equations," *Trans. London Philosoph. Soc. (A)*, vol. 209, pp. 415–446, 1909.

[25] A. E. Hoerl and R. W. Kennard, "Ridge regression: Biased estimation for nonorthogonal problems," *Technometrics*, vol. 42, no. 1, p. 8086, 1970.

[26] A. N. Tychonoff, "On the stability of inverse problems," *Doklady Akademii Nauk SSSR*, vol. 39, no. 5, pp. 195–198, 1943.

[1]The views, opinions, and/or findings contained in this article/presentation are those of the author/presenter and should not be interpreted as representing the official views or policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the Department of Defense.

[27] S. Y. Kung and P. Wu, "On efficient learning and classification kernel methods," *Proc. ICASSP, Kyoto*, March 2012.

[28] M. Obaidat and B. Sadoun, "Verification of computer users using keystroke dynamics," *IEEE Trans. Syst., Man, Cybern. B*, vol. 27, no. 2, pp. 261–269, Apr 1997.

[29] F. Bergadano, D. Gunetti, and C. Picardi, "User authentication through keystroke dynamics," *ACM Trans. Inf. Syst. Secur.*, vol. 5, no. 4, pp. 367–397, nov 2002.

[30] Y. Sheng, V. Phoha, and S. Rovnyak, "A parallel decision tree-based method for user authentication based on keystroke patterns," *IEEE Trans. Syst., Man, Cybern. B*, vol. 35, no. 4, pp. 826–833, Aug 2005.

[31] D. Hosseinzadeh and S. Krishnan, "Gaussian mixture modeling of keystroke patterns for biometric applications," *IEEE Trans. Syst., Man, Cybern. C*, vol. 38, no. 6, pp. 816–826, Nov 2008.

[32] K. S. Killourhy and R. A. Maxion, "Comparing anomaly-detection algorithms for keystroke dynamics," in *Proc. 39th Annu. Int. Conf. Dependable Syst. Networks*, ser. DSN 2009. Los Alamitos: IEEE Computer Society Press, 2009, pp. 125–134.

[33] A. Alsultan and K. Warwick, "Keystroke dynamics authentication: A survey of free text methods," *Int. J. Comput. Sci. Issues*, vol. 10, no. 1, pp. 1–10, July 2013.

[34] F. Monrose and A. Rubin, "Authentication via keystroke dynamics," in *Proc. 4th ACM Conf. Comput. Commun. Security*, ser. CCS '97. New York, NY, USA: ACM, 1997, pp. 48–56.

[35] A. Ahmed and I. Traore, "Biometric recognition based on free-text keystroke dynamics," *IEEE Trans. Cybern.*, vol. 44, no. 4, pp. 458–472, April 2014.

[36] M. Villani, C. Tappert, G. Ngo, J. Simone, H. Fort, and S.-H. Cha, "Keystroke biometric recognition studies on long-text input under ideal and application-oriented conditions," in *Conf. Comput. Vision Pattern Recognition Workshop, CVPRW '06*, June 2006.

[37] H. Davoudi and E. Kabir, "A new distance measure for free text keystroke authentication," in *14th Int. CSI Comput. Conf. (CSICC 2009)*, Oct 2009, pp. 570–575.

[38] S. Park, J. Park, and S. Cho, "User authentication based on keystroke analysis of long free texts with a reduced number of features," in *2nd Int. Conf. Commun. Syst. Networks Applicat. (ICCSNA 2010)*, vol. 1, June 2010, pp. 433–435.

[39] S. Singh and K. V. Arya, "Key classification: A new approach in free text keystroke authentication system," in *3rd Pacific-Asia Conf. Circuits, Commun. Syst. (PACCS 2011)*, July 2011, pp. 1–5.

[40] M. Curtin, M. Villani, G. Ngo, J. Simone, H. S. Fort, and S. h. Cha, "Keystroke biometric recognition on long-text input: A feasibility study," in *Proc. Int. Workshop Sci Comp/Comp Stat (IWSCCS 2006), Hong Kong*, 2006.

[41] D. Gunetti and G. Ruffo, "Intrusion detection through behavioral data," in *Advances in Intell. Data Anal.*, ser. Lecture Notes in Computer Science, D. Hand, J. Kok, and M. Berthold, Eds. Springer Berlin Heidelberg, 1999, vol. 1642, pp. 383–394.

[42] R. Janakiraman and T. Sim, "Keystroke dynamics in a general setting," in *Advances in Biometrics*, ser. Lecture Notes in Computer Science, S.-W. Lee and S. Li, Eds. Springer Berlin Heidelberg, 2007, vol. 4642, pp. 584–593.

[43] J. Hu, D. Gingrich, and A. Sentosa, "A k-nearest neighbor approach for user authentication through biometric keystroke dynamics," in *IEEE Int. Conf. Commun., ICC '08.*, May 2008, pp. 1556–1560.

[44] A. A. E. Ahmed, I. Traor, and A. Ahmed, "Digital fingerprinting based on keystroke dynamics." in *HAISA*, N. L. Clarke and S. Furnell, Eds. University of Plymouth, 2008, pp. 94–104.

[45] A. Messerman, T. Mustafic, S. Camtepe, and S. Albayrak, "Continuous and non-intrusive identity verification in real-time environments based on free-text keystroke dynamics," in *Int. Joint Conf. Biometrics (IJCB 2011)*, Oct 2011, pp. 1–8.

[46] J. M. Chang, C. C. Fang, K. H. Ho, N. Kelly, P. Wu, Y. Ding, C. Chu, S. Gilbert, A. E. Kamal, and S. Y. Kung, "Capturing cognitive fingerprints from keystroke dynamics," *IT Professional*, vol. 15, no. 4, pp. 24–28, July-August 2013.

[47] P.-Y. Wu, C.-C. Fang, J. Chang, S. Gilbert, and S. Kung, "Cost-effective kernel ridge regression implementation for keystroke-based active authentication system," in *IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP 2014)*, May 2014, pp. 6028–6032.

[48] T.-T. Frieß and R. F. Harrison, "A kernel-based adaline," in *ESANN 1999, 7th European Symp. Artificial Neural Networks, Bruges, Belgium, April 21-23, 1999, Proceedings*, 1999, pp. 245–250.

[49] Y. Engel, S. Mannor, and R. Meir, "The kernel recursive least squares algorithm," *IEEE Trans. Signal Process.*, vol. 52, pp. 2275–2285, 2003.

[50] J. Kivinen, A. J. Smola, and R. C. Williamson, "Online learning with kernels," *IEEE Trans. Signal Process.*, vol. 52, no. 8, pp. 2165–2176, 8 2004.

[51] W. Liu, P. P. Pokharel, and J. C. Principe, "The kernel least-mean-square algorithm," *IEEE Trans. Signal Process.*, vol. 56, no. 2, pp. 543–554, 2 2008.

[52] C. Saunders, A. Gammerman, and V. Vovk, "Ridge regression learning algorithm in dual variables," in *Proc. 15th Int. Conf. Mach. Learning*, ser. ICML '98. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1998, pp. 515–521.

[53] C. C. Chang and C. J. Lin, "LIBSVM: A library for support vector machines," *ACM Trans. Intell. Syst. and Tech.*, vol. 2, pp. 27:1–27:27, 2011, software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

[54] A. Vedaldi and A. Zisserman, "Efficient additive kernels via explicit feature maps," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 3, pp. 480–492, March 2012.

[55] S. Phonphitakchai and T. Dodd, "Stochastic meta descent in online kernel methods," in *6th Int. Conf. Elect. Eng./Electron., Comput., Telecommun. Inform. Tech. (ECTI-CON 2009)*, vol. 02, May 2009, pp. 690–693.

[56] C. Richard, J. Bermudez, and P. Honeine, "Online prediction of time series data with kernels," *IEEE Trans. Signal Process.*, vol. 57, no. 3, pp. 1058–1067, March 2009.

[57] Z. Chair and P. K. Varshney, "Optimal data fusion in multiple sensor detection systems," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 22, pp. 98–101, Jan 1986.

[58] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: Synthetic minority over-sampling technique," *J. Artificial Intell. Research*, vol. 16, pp. 321–357, 2002.

[59] G. M. Weiss, "Mining with rarity: A unifying framework," *SIGKDD Explor. Newsl.*, vol. 6, no. 1, pp. 7–19, jun 2004.

[60] G. E. A. P. A. Batista, R. C. Prati, and M. C. Monard, "A study of the behavior of several methods for balancing machine learning training data," *SIGKDD Explor. Newsl.*, vol. 6, no. 1, pp. 20–29, 2004.

[61] B. Zadrozny and C. Elkan, "Learning and making decisions when costs and probabilities are both unknown," in *Proc. 7th ACM SIGKDD Int. Conf. Knowledge Discovery Data Mining*, ser. KDD '01. New York, NY, USA: ACM, 2001, pp. 204–213.

[62] C. K. I. Williams and M. Seeger, "Using the nyström method to speed up kernel machines," in *Advances in Neural Inform. Process. Syst. 13*, T. Leen, T. Dietterich, and V. Tresp, Eds. MIT Press, 2001, pp. 682–688.

[63] M. Espinoza, J. Suykens, and B. Moor, "Fixed-size least squares support vector machines: A large scale application in electrical load forecasting," *Computational Manag. Sci.*, vol. 3, no. 2, pp. 113–129, 2006.