Text Dependent Writer Verification using Boosting

Sachin Gupta and Anoop Namboodiri
International Institute of Information Technology, Hyderabad, India
sachin_g@students.iiit.ac.in, anoop@iiit.ac.in

Abstract

Text-dependent writer verification systems are preferred over text-independent systems due to the accuracy they achieve with small amount of data. However, text-dependent systems are prone to forgery. This paper proposes a novel boosting based framework for writer-specific text generation to increase the accuracy and a method of text variation to make the system robust to forgery. The approach is able to achieve error rates of 5% *with just* 6 *words as compared to random(*11%*) or most discriminative(*22%*) primitive selection methods on a dataset containing* 20 *writers. Boosting based text selection also provides the flexibility to incorporate text variation across multiple authentications, which in turn makes the system robust to forgery.*

Keywords: Writer Verification, Boosting, Text Generation

1 Introduction

Text dependent methods for writer verification offers higher discriminative power using lesser amount of data, as compared to text-independent methods [1]. However, text-dependent systems also introduce some additional challenges to a writer identification or verification system. The primary problem is that one needs to have handwritten data that is labeled with the corresponding text. In the case of writer verification, this is not so difficult, as the user can be asked to write a pre-determined piece of text, which in turn can be aligned with the collected data. However, the use of pre-determined text increases the chances of forgery, as an impostor could practice writing the specific text to make it look like that of the genuine user. The problem has been studied extensively in the case of signature verification, which is a special case of text-dependent writer verification. In the case of generic writer verification an effective solution is to vary the verification text across authentications.

Another challenge is the determination of appropriate verification text for each writer, as the discrimination power of different characters or words vary across writers [1]. The use of appropriate discriminating text is critical as it minimizes the amount of data required for accurate authentication, making the system usable in practical situations such as access control or ATMs. The need for using writer-specific discriminating text, when compounded with the requirement to vary the text across authentications to reduce forgery, makes the problem of generating the authentication text a difficult one.

In short, one needs to be able to generate writer-specific discriminative authentication text, that varies over time for effective and accurate writer verification. Moreover, the authentication algorithm should be able to utilize the text generated to perform text-dependent verification. Such a system can combine the advantages of both text-independent (*robust to forgery)* and text-dependent systems *(high accuracy with less data)* to verify the writer of the system.

Specifically, we look at the problems related to a practical verification system for low security and access control applications in the civilian domain, as opposed to high security or forensic applications. In case of low security civilian applications, the primary requirements are *ease of use* and *low false rejection rate*, i.e., genuine user should not be inconvenienced with frequent rejects. Another major difference that arises from the domain of use is the control over data collection. We can instruct the user to provide online data, which may not be possible in case of forensic applications. In this paper, we propose a framework to generate writer-specific test data at verification stage, which also suits the requirements of the practical verification system. We do not attempt a detailed survey of the work in writer identification due to space constraints. Comprehensive surveys of the related work are available in Schomaker et al.[9] and Bunke et al.[7].

2 Generation of Authentication Text

Based on our discussion on practical writer verification systems, the major problem of verification is to identify a combination of primitives of handwriting (strokes, characters, words, etc.) that can discriminate a specific writer from other writers. At the same time, variation of the text is also necessary as a fixed text increases the

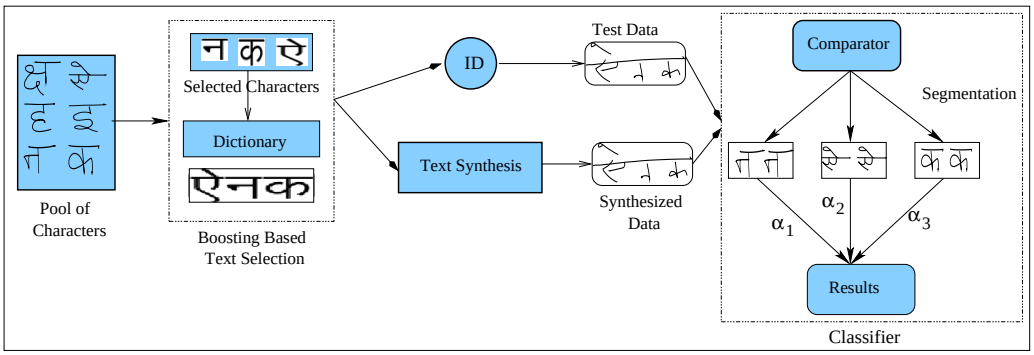


Figure 1. Writer verification framework for low security access control applications.

chances of forgery. We will now look at the algorithm to generate writer-specific text and introducing variations in the text while preserving the discrimination power.

Handwriting is a behavioral biometric trait. The discriminating power of different primitives (strokes, characters, words, etc.) of the handwriting vary for individuals and a single primitive of handwriting is not discriminating enough to accurately identify a particular writer [3]. Hence, generation of writer-specific text is essential to achieve high accuracy with a small amount of text.

Given a set of primitives (strokes, characters, or words), none of which is individually efficient enough to discriminate the writer from other writers (i.e., weak classifiers), the problem of text generation is to select the most discriminating and compact text for a specific writer. This can be done using an automatic feature selection method. Bunke et al. have proposed a feature selection based offline writer verification algorithm [8]. Most of the feature selection methods search for a globally optimal set of weights for the primitives. However, as we noted in case of handwriting, there are many possible subsets of primitives and associated weights that can achieve a high accuracy. For text variation, we need a feature selection method that can obtain these different combinations efficiently. In this paper, we present a boosting based feature selection method for writer-specific text generation. Boosting [6] is a general method to combine "weak" PAC learning algorithms into one with arbitrarily high accuracy. We use Adaboost [6], an efficient (greedy) boosting algorithm, which selects weak classifiers (primitives) at each stage based on the previously selected classifiers. Such an approach is well suited for text variation, as selection of a particular primitive at a given stage will automatically result in the selection of complementary classifiers at later stages to achieve high accuracy. Adaboost, have also been used previously as a feature selection method in other domains (e.g., face detection [10]).

2.1 Writer-Specific Text Generation

The training stage of the algorithm learns the discriminative power of the individual primitives by building classifiers using each primitive. The generation of verification text happens in two stages: i) Primitive selection and ii) Text generation. The boosting based algorithm selects primitives, one at a time, based on the average discrimination between pairs of users. Once the primitives are decided, a dictionary module is used to generate meaningful words out of the primitives.

Figure 1 shows the primitive selection and text generation algorithm. At each iteration, the algorithm selects a primitive that increases the separation between the claimed writer and the current most similar writers. An important advantage of the adaboost algorithm is that it automatically selects complementary primitives to those selected by the previous iterations, while providing better generalization than other classifier combination mechanisms. The key insight is that generalization error is related to margin of the examples and the adaboost achieves large margins rapidly. In the next section, we explain the cascaded classifier structure in order to make the system faster and less complex.

2.1.1 Cascaded Classifier

To increase the efficiency of the verification process, we employ a cascaded classifier that eliminates unlikely writers at each stage. Each stage of the cascade consists of a boosted classifier that separates a set of writers from the claimed one. After each stage in the cascade, we eliminate those writers from consideration, of whom we are confident of not being the user under consideration. The structure of the cascaded classifiers is essentially that of a "degenerate decision tree" and is related to the work of Fleuret & Geman [2]. At each stage in the cascaded structure, a threshold is selected, which decides the writers to be rejected. The threshold is selected such that the verification system is biased towards accepting the writer to be verified. As the number of writers decreases over the

stages, the number of samples that require the stage also decreases, which in turn, makes system fast. Once a writer is rejected by any one of the stages, he will not be considered for design of any of the subsequent stages. The overall algorithm is given below:

Algorithm: Boosting based text generation algorithm

Require: ID of the writer to be verified, weak hypothesis h_j

1: Initialize $D_1(i) = 1/n$, where $i = 1, \ldots, m$, where n is the number of writers in the system
2: writer-List = $\{1, \ldots, n\}$
3: **while** $|writer - List| > 1$ **do**
4: **while** $Nowriterisrejected$ **do**
5: Select weak learner, h_t^k, such that $\epsilon_k = \sum_i D_t(i)\theta_i^{ID}$ is maximum. θ_i^{ID} represents the pairwise discriminating power of the classifier h^k for and calculated during the enrollment stage
6: Calculate: $\alpha_t = \log\left(\frac{1-\epsilon_k}{\epsilon_k}\right)$
7: Update: $D_{t+1}(i) = \frac{D_t(i)exp\left(\alpha_t\theta_i^{ID}\right)}{Z_t}$ where Z_t is a normalization factor (chosen so that D_{t+1} will be distribution).
8: Compute threshold Th for the stage using within-writer distance.
9: if ($\xi_i > \mathrm{Th}$) Reject the writer
10: **end while**
11: **end while**

2.2 Text Variation

The boosting-based feature selection algorithm will select a highly discriminative set of primitives for each user, which in turn is used for text generation. However, the discriminating power of primitives do not change during the testing phase, and the same set of primitives and text will be selected for every authentication. The problem of forgery will still remain although the text selected is specific for each writer. In order to overcome this limitation, we introduce randomness into the primitive selection stage. Each primitive that is chosen in an iteration of boosting can be selected or rejected based on random number. The boosting based feature selection method is flexible enough to select subsequent primitives so that the discriminative power of the selected set of primitives for the user is high. As the primitives that are chosen vary over authentications, the corresponding text generated by the dictionary based unit also varies considerably.

We have assumed the text generated to be a set of words that covers the primitives that are chosen. One could employ language models to create meaningful sentences.

3 Writer Verification

With the primitive selection and text generation algorithms, we are able to generate text that is discriminative and specific to each writer, while introducing variations across authentications. Pre-defining a fixed set of verification texts for a user limits the amount of variability that can be introduced for each writer. In order to solve this problem, the discriminating text generation phase is delayed until authentication. Each time a user tries to authenticate his identity, the text generation phase is invoked to obtain a new verification text, and the writer is asked to write the corresponding text.

The verification algorithm consists three steps: 1) Separation of primitives from the written text, 2) Comparison of the written primitives with those obtained during the enrollment phase, and 3) Combination of the primitive classifiers to obtain the verification result.

The first step requires the segmentation of words, characters, or sub-characters from a given piece of handwritten text. This is a challenging problem in free-form writing. However, in our case, we know the text being written and hence it can be formulated as an alignment problem using text synthesis. We have used the approach proposed by Kumar *et al.* [4] to accomplish the annotation along with segmentation. The primary idea here is to synthesize the verification text in the claimed writer's handwriting and align it with the verification data.

Step 2 is straight forward as the primitive based classifiers can be trained during the enrollment phase. For combining the results from the individual primitive classifiers, we fall back on the boosting based text generation algorithm. Note that the primitive selection algorithm also provides us with the weights and threshold for each primitive and cascade stage to carry out the verification. Hence, once the primitives are segmented out, we can use the cascaded classifier developed during text generation to compute the final verification decision. The actual verification proceeds as follows:

Let each stage in the cascaded classifiers denoted by W_i, where $i = 1, \ldots n$. n is the number of cascaded stages in the classifier. Final hypothesis H is given by:

$$H(x) = \prod_i (W_i < \vartheta_i) \tag{1}$$

W_i is the score of the writer x, for i^{th} cascade and ϑ_i is the threshold for i^{th} cascade calculated during text generation phase. Rejection at any stage will also reject the claim. Score W_i of each cascaded classifier is calculated as the combination of various weak hypotheses selected at each selection stage. let h_j be the j^{th}weak hypothesis. Then W_i is given by:

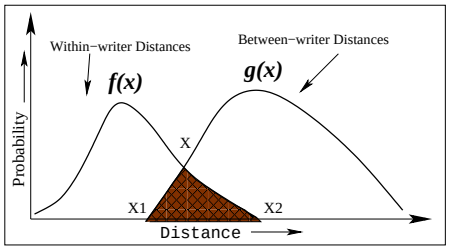


Figure 2. Discriminating power of a primitive is proportional to the area of intersection.

$$W_i = \sum_j \alpha_j h_j, \tag{2}$$

where α_j is the relative importance or weight given to j^{th} weak classifier computed during adaboost based text generation phase and $h_j(X)$ is the hypothesis generated by j^{th} classifier within a single stage.

3.1 Enrollment Phase

In traditional writer verification process, enrollment phase is to identify the threshold of between-writer distances to within-writer distances. In the framework, as text generation phase is delayed till authentication phase. The calculation of discriminating power and training of synthesis phase is done during the enrollment phase.

3.1.1 Primitive Discrimination

Discrimination is defined as the degree of separation of within- and between-writer distances between a pair of writers. Discrimination power is calculated for each primitive between each pair of writers.

The discriminatory power of a primitive is defined as:

$$D_{ij}(w) = 1 - \left(\int_{X_1}^{X} g(x) + \int_{X}^{X_2} f(x) \right), \tag{3}$$

where $D_{ij}(w)$ is the discriminatory power of word w for writers W_i and W_j and $f(x)$ and $g(x)$ are the distributions of within writer and between writer distances. So, the discriminating power of words, essentially, is inversely proportional to the overlap between distribution(see Figure 2). The more the overlap between distributions, the less the discriminating power and vice versa.

3.2 Feature Extraction

In our experiment, we have used words as the basic unit (primitive). Each online word is represented as the set of strokes, which in turn is a sequence of points. For experimental purposes, two different methods are used for comparison between strokes: dynamic time warping(DTW [5]) and directional features.

(1) *DTW Matching:* DTW based matching is a natural choice for the word distance as the number of strokes in similar words are not the same. Moreover, it provides us an efficient method to compare different length feature vectors.

(2) *Directional features:* It has been proven before that direction based features [9] possess a lot of individuality information. For this set of experiments, the curvature of the strokes is calculated at each point and grouped into 12 bins. Euclidean distance between these fixed dimensional feature vectors is used for distance estimation.

Once the distance between all pairs of strokes are calculated, dynamic time warping is used to calculate the distance between words. Since, the order and the number of strokes possess individuality information about the writer, DTW can be used as an effective classifier, by varying penalty for misalignment.

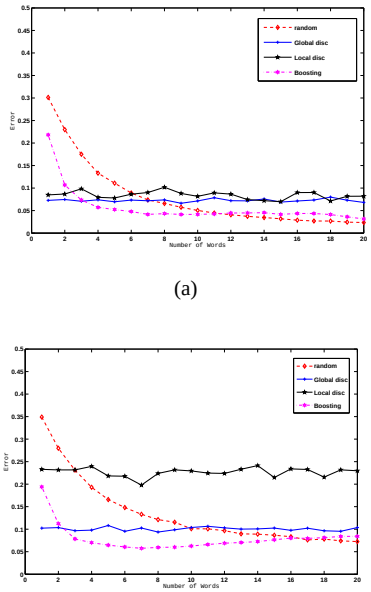
4 Experimental Results and Analysis

For the purpose of experiments, the data is collected from different writers using an online data capturing tablet. The data was collected in Hindi and English scripts. Hindi data is collected from 10 users and English data is collected from 20 users. Each person have written 20 distinct words, $10 - 12$ times each. All experimental results reported here were from 3-fold cross validation.

The Algorithm is compared with other feature/text selection methods such as random and discriminating power based selection methods, in order to decide the effectiveness of the approach. In random selection, the primitives are selected randomly from the given database and are given equal weights. In the discriminating power based selection method, words are selected based on their discriminating power (for the given set of writers). Two variants of discriminating power based selection methods are used, i.e. discriminating power of primitives taking all the writers together (global discriminating power) and discriminating power of the words for individual writers (local discriminating power).

Figure 3 compare the accuracies of different primitive selection methods for Hindi script. It is evident from the graph that accuracies of boosting based randomized method are quite comparable to discriminating power based primitive selection and quite higher than random selection with small number of words. For only 7 words, error rates of boosting based selection method is around 5%, which is lower than other selection methods. Figure 4 compares the False acceptance (FAR) and false rejection (FRR) rates for directional features.

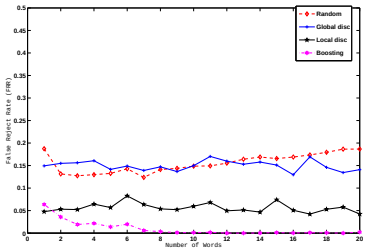
It is evident from the FRR graph 4, that the performance of individual discrimination based selection is better than global discrimination based selection method. The is due to the fact that in the threshold is selected based



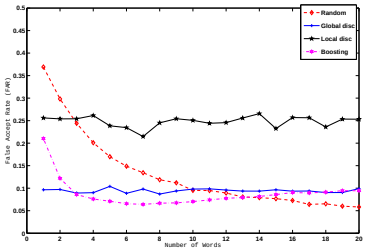
(a)

(b)

Figure 3. Error rates for DTW and Directional features with increasing number of words.



(a)



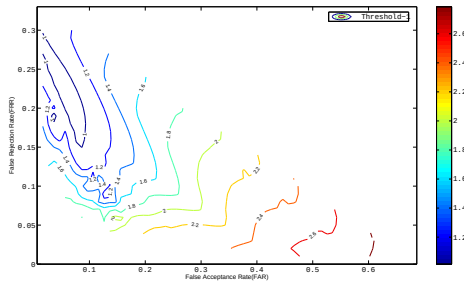
(b)

Figure 4. (a)FRR, and (b) FAR for different text selection methods.

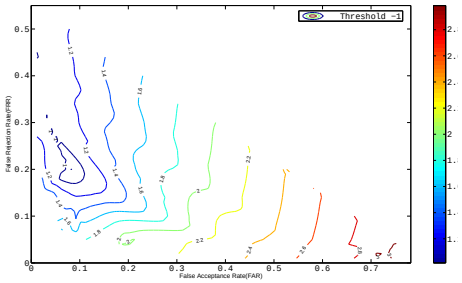
on a specific classifier and thus it performs better for the individual writer and provide lower FRR. However, FAR are higher for local discriminating power based selection as in these cases threshold is selected such that it is biased towards accepting the genuine writer. This is done as in case of boosting too threshold is biased towards the writer. Boosting based method selects primitives dynamically based on the individual writers and thus performs better than other approaches. Boosting also provides better generalization performance and this is evident from the figure, as FRR rates drops as number of stages increases.

The traditional writer verification system uses single threshold (of within and between-writer distances) for the authentication. The cascaded classifiers based proposed system uses two thresholds based on both positive and negative samples. Threshold-1 (th-1), which is calculated based on the positive samples (within-writer distances), affects the FRR of the system. This threshold is usually selected such that it accepts all the positive training samples to keep the FRR low. However, in turn, the FAR will be high as with a relaxed th-1, there are more chances of impostors to be passed through at every stage. Threshold-2 (th-2) is chosen based on the negative samples and effectively controls FAR. It helps in deciding, whether to reject a writer from consideration from the next stage in the cascade. Both thresholds affects the performance of the system. For the sake of experiments, (during training) th-2 is taken as the percentage of the negative samples below first threshold. In other words, if th-2 is selected to be 20%, then we reject all the writers (from consideration in subsequent stages), who have less than 20% of the samples below th-1. This essentially means that those writers are already sufficiently different from the writer under consideration. For the experimental purpose, th-2 was varied from 5% to 50% with the step size of 5 and th-1 was varied as a multiple of the initial threshold from 1 to 3 with the step size of 0.25. Figure 5 below shows the performance of the system for different values of the two thresholds. As seen from the above graphs, FAR increases with increasing th-1 for various values of th-2 (as expected). At the same time, FRRs are quite low with higher values of th-1. At higher value of th-1 there is a higher chance for imposters to be allowed to pass through the system.

Another major concern with biometrics based verification systems is that of scalability. As the number of writers increases, the performance of the system often decreases. However, in the cascaded boosting based method, performance of the system is not considerably affected by the increasing number of writers (see Figure 6). As evident from the graph 6, the error term is decreasing with the increase in the number of writers. This is due to the generalization capability of boosting based systems. At the same time, as the number of writers increases, the number of the cascaded stages also increases. More the number of stages, the writer have to pass through more rigorous testing. For less number of writers, number of stages will

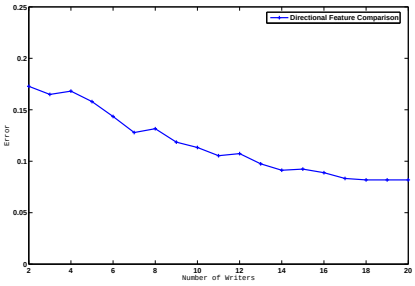


(a)

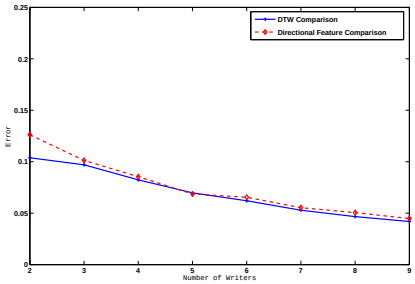


(b)

Figure 5. ROC curves with varying threshold-1 for directional features (a) Hindi script, (b) English Script.



(a)



(b)

Figure 6. Error Rates as the function of Number of writers (a) English Script (b) Hindi Script.

be less and since the system is biased towards accepting the writers rather than rejecting, the false acceptance rates will be higher. As the number of writer increases, effect due to biasing reduces and makes the system more accurate.

5 Conclusions and Future Work

We proposed an effective automatic writer specific text generation mechanism that can achieve high accuracy with limited amount of text. Our algorithm is flexible enough to generate different text for multiple authentication, while retaining the discriminating power, making the system forgery resistant. The system is designed specifically for low security access control and civilian applications as false rejection rates can be controlled independently. Experimental results demonstrate the effectiveness of the proposed boosting based framework. The approach is also applicable to other biometric modalities such as speech and keystroke dynamics.

References

[1] B. Z. Catalin I. Tomai and S. N. Srihari, "Discriminatory Power of Handwritten Words for Writer Recognition", *International Conference on Pattern Recognition*, 2004, volume 2, pp 638–641, Cambridge, UK.

[2] F. Fleuret and D. Geman, "Coarse-to-Fine Face Detection", *International Journal of Computer Vision*, 41(1/2):85–107, 2001.

[3] R. Huber and A. Headrick, *Handwriting Identification: Facts and Fundamentals*, Boca Roton, CRC Press, 1999.

[4] A. Kumar, A. Balasubramanian, A. Namboodiri and C. Jawahar, "Model-based Annotation of Online Handwritten Datasets", *International Workshop on Frontiers in Handwriting Recognition*, October 2006, pp 9–14, La Baule, France.

[5] H. Sakoe and S. Chiba, "Dynamic Programming Algorithm Optimization for Spoken Word Recognition", *IEEE transactions on Acoustics, Speech, and Signal processing*, Assp-26(1), Feb 1978.

[6] R. E. Schapire, "Theoretical Views of Boosting", *Lecture Notes in Computer Science*, 1572:1–10, 1999.

[7] A. Schlapbach and H. Bunke, "A Writer Identification and Verification System Using HMM Based Recognizers", *Pattern Analysis & Applications*, 10(1):33–43, 2007.

[8] A. Schlapbach, V. Kilchherr and H. Bunke, "Improving Writer Identification by Means of Feature Selection and Extraction", *International Conference of Document Analysis and Recognition*, 2005, pp 131–135.

[9] L. Schomaker, K. Franke and M. Bulacu, "Using codebooks of fragmented connected-component contours in forensic and historic writer identification", *Pattern Recognition Letters*, 28(6):719–727, 2007.

[10] P. A. Viola and M. J. Jones, "Robust Real-Time Face Detection", *International Journal of Computer Vision*, 57(2):137–154, 2004.