Offline Signature Verification using hybrid DRT-PCA and DCT methods with SVM Classifier

Robert Neff
Stanford University
rneff@stanford.edu

Nathan Butler
Stanford University
nbutler@stanford.edu

Abstract

Handwritten signatures play an important role in providing authentication of one's identity on legal, financial or other hard copy documents. As such, automated verification of a signature's legitimacy is a paramount task in protecting and proving a person's name. We propose two methods for global feature extraction, namely: hybrid discrete radon transform (DRT) with principle component analysis (PCA) and discrete cosine transform (DCT), after which features are classified with a support vector machine (SVM). Using the La Trobe database composed of genuine signatures and skilled forgeries for two users, our methods achieve equal error rates (EER) of 13% for DRT & PCA method and 14% for DCT method with 250 training samples sizes. Using the Caltech Vision Group database of hundreds of users with genuine signatures and casual forgeries, our system achieves EER of 18% and 22% with 250 training samples for above mentioned methods respectively. These signatures were originally stored in online form, then converted to static offline signature images.

1. Introduction

In the modern age there are two modes for providing signatures, namely offline and online. In the offline case, signatures are provided on paper via traditional writing tools and then scanned as two-dimensional images for inspection. The online system takes signatures from electronic media, such as tablets, and records not only the signature images themselves but also dynamic writing components, like acceleration and pressure of the stylus. As the online system is far less limited in the amount of data available for verification than its offline counterpart, forgeries become difficult. Physical documents, however, remain critically used in many fields, especially in non point-of-purchase (POP) transactions.

Categorical signature discrimination systems can focus on either verification or recognition tasks. Signature verification seeks to assess whether a signature belongs to a certain class (or writer) with a true-or-false claim. Conversely, signature recognition seeks to assign a class to a specific signature.

In this paper, we examine offline signature verification, as verifying the signee of a check (among other items) remains an ongoing problem, and signature verification poses an inexpensive option to reduce intrusive manual screening costs. As such, we consider the skilled and casual forgery categories of signatures in our system. A skilled forgery is the result of a forger who has unrestricted access to samples of the original writer's signature. A casual forgery occurs when the forger is made aware of the person's name but not provided with an image of their signature - this often leads to stylistic differences. We forego the random forgery, which consists of any word or scribble written by another writer.

Our signature verification pipeline consists of three key stages: image pre-processing, feature extraction and encoding, and binary classification. This setup if fairly common to signature verification systems and implementation was inspired by that Ooi et al. [12], whose paper we we implemented before introducing our own input.

Following their approach, our image pre-processing step ensures maximal normalization of the incoming scanned image before the next step. Features are then extracted via DRT and subsequent PCA processing of the sinogram per specifications similar to Ooi et al. [12]. We introduce our own input by implementing a second feature extraction option in the DCT, where the lowest frequency coefficients of the DCT transformed image function as an alternative to the DRT principle components for comparison. Both approaches yield features classified under global features, as they describe the entire image rather than looking to local components such as stroke, slant and pressure. This provides a more universal and robust system for signature differences. Finally, unlike Ooi et al. [12] who used a PNN, we use SVM learning to partition the features into genuine and forged categories, thus being able to easily discriminate the authenticity of a signature for a particular person.

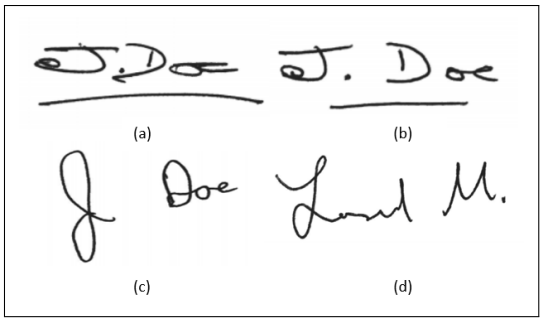

Figure 1. Example of a (a) genuine signature, (b) skilled forgery, (c) casual forgery, and (d) random forgery for the writer "J. Doe."

2. Related Work

Many researchers have investigated the signature verification problem space in the past few decades. As technology and techniques have developed, the focus of most papers has transitioned from dealing with random and casual forgeries to primarily skilled forgeries. As such we review some of these works in this section.

Özgündüz et al. [11] propose an SVM classification approach for signature verification and recognition using global, directional and grid features. By training 8 positive (genuine) and 82 negative (forged) signatures, of which 78 were random and 4 skilled, for each person in their dataset they report a true classification ratio of 0.95. They compare this to an artificial neural network (ANN) back propagation approach which only yields a true classification ratio of 0.75, is more difficult to implement and has less optimized runtime.

Coetzer et al. [4] proposed a signature verification system using the DRT for feature extraction and a hidden Markov model (HMM) for classification. On the Stellenbosch database of 924 signatures from 22 writers, their method achieved an equal error rate (EER) of 18% for skilled forgeries and EER pf 4.5% on casual forgeries given 10 training signatures per writer. Following that on Dolfing's data set of 4800 signatures from 51 writers they achieve EER of 12.2% for skilled forgeries only, using 15 training samples per writer.

Chandra et al. [3] propose geometric global features, such as area, centroid and kurtosis (among others) in combination with ANN to tackle the signature verification problem. Training on 180 signatures from the MCYT offline signature corpus and testing on 18 users they achieve a false acceptance rate (FAR) of 10.62% and false recognition rate (FRR) of 10.91% for a total accuracy of 89.24%.

Bhattacharya et al. [2] propose using a pixel matching technique (PMT) to verify user signatures in their database. After multiple pre-processing stages, they compare two binary signatures images pixel-by-pixel, increasing and decreasing a similarity counter throughout the process. Using this approach they report FAR and FRR rates of 6% and 12% respectively.

Recently, Ooi et al. [12] proposed a scheme to extract signature features using DRT and subsequently PCA to only examine the most significant features and so reduce the amount of data stored and a probabilistic neural network (PNN) to classify results. Testing on their own database of 100 signatures split equally into casual and skilled forgeries, they report ERR of 1.51%, 3.32% and 13.07% for random, casual and skilled forgeries respectively. Using the MYCT signature databse they achieve an EER of 9.87% using just 10 training samples.

Jana et al. [5] propose using global ratio and positioning features in addition to bounding boxing and cropping signature images to match features based on distance thresholds. Query images that surpass the feature threshold are then classified as forgeries. By training on each of 7 users with 10 valid signatures, then testing each user with 5 genuine and 10 forged signatures they report average FRR of 2.86% and FAR of 17.14% across all user sets. Furthermore, their yield an average accuracy is 87.61%, which they note, suffers in the case of skilled forgeries.

3. Our Approach

3.1. Image Pre-Processing

Scanned signatures can contain noise due to blemishes and irregularities on the physical paper and/or the scanning hardware itself. To mitigate this noise, the image is converted from RBG to gray-scale, then undergoes median-filtering, by which each pixel is set to the median of the pixels in its designated neighborhood. Median-filtering is preferable to mean-filtering as it keeps the overall image sharp in addition to ridding it of unwanted noise. The final step binarizes the image based on a predetermined threshold further reducing overall noise. An additional consequence of this step is that the image now requires far less space for storage, making computations faster. These changes can be seen from Figure 2 to 3.

Figure 2. A potential signature image before the pre-processing step [7].

Figure 3. The same signature image after the pre-processing step.

3.2. Discrete Radon Transform (DRT)

The DRT represents projections of the original signature image at different angles, a process often used in medical imaging technology. The DRT of each image is calculated as follows. First, each image is assumed to contain N pixels, with each pixel having intensity I_i, $i = 1, ..., N$. The DRT is calculated using k non-overlapping beam projections into the image per angle, with Θ angles in total. Each beam sums the intensity of the pixels that fall within it per some interpolated weight $w_i^{(j\alpha)}$ for the given angle α and beam j, where $w_i^{(j\alpha)} = 0$ if the pixel is not in the beam. Each beam sum $R^{(j\alpha)}$ of the resulting sinogram (the DRT of the image) and can be expressed as:

$$R^{(j\alpha)} = \sum_{i=0}^{N} w_i^{(j\alpha)} I_i \tag{1}$$

for $j = 1, ..., k$ and $\alpha = 1, ..., \Theta$.

Rather than computing the interpolation values for each pixel along a specified beam directly, the image can be rotated via image warping functions that automatically performs the desired interpolation step. Each beam sum for a given rotation of the image is then the cumulative intensity of the values in each column. Having each beam correspond to a column in the padded image requires that the beam width be 1 pixel. To get the desired number of beams then, the image is scaled down or up such that the maximal dimension has size k.

Since images tend not to be circular in shape, each image is padded before rotation such that no rotated pixels from the original image ever fall outside the padded region. A consequence of this method is that some beam sums will fall in the padded region and always contribute a zero. To fix this each column of the resulting sinogram, i.e. $R^{(j\alpha)}$ for some α, has its zero values decimated and is resized (shrunk or expanded) to a new length d (see Fig. 4). To ensure that noise levels due to interpolation are minimal, the value of d is selected such that $d < k$. This process also helps ensure shift invariance, as the signature may not be localized in the center of the image.

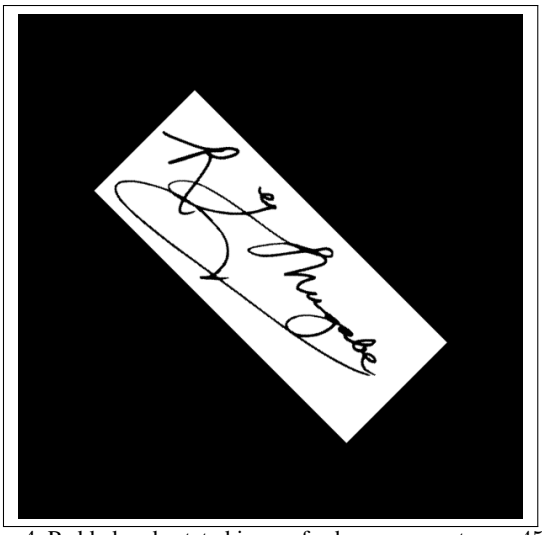

Figure 4. Padded and rotated image for beam sums at $\alpha = 45°$.

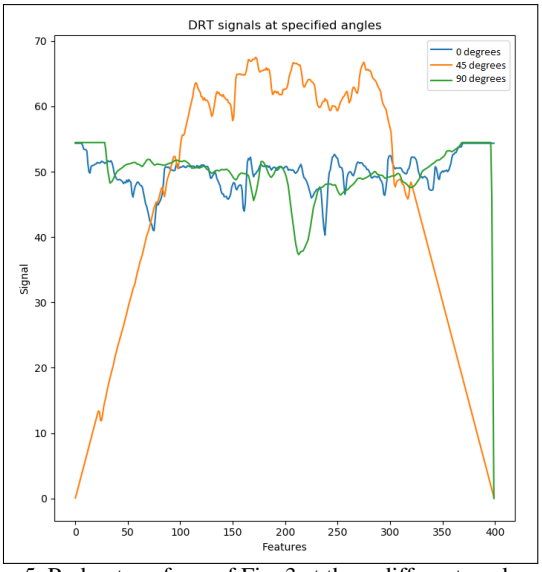

Figure 5. Radon transform of Fig. 3 at three different angles.

Further post-processing steps ensure scaling and rotational invariance. Scale invariance in the DRT must be achieved in both the direction perpendicular to the beam scanning direction and that parallel to it. Perpendicular scale invariance is enforced by the aforementioned zero-value decimation and resizing of each column. Parallel scale invariance is enforced via normalizing each column projection. Rotational invariance must be achieved as the input signature may not be written along a perfect horizontal line in the image. To negate this problem, the DRT can be computed for angles that range from $0°$ to $360°$, i.e. $\Theta = 360$.

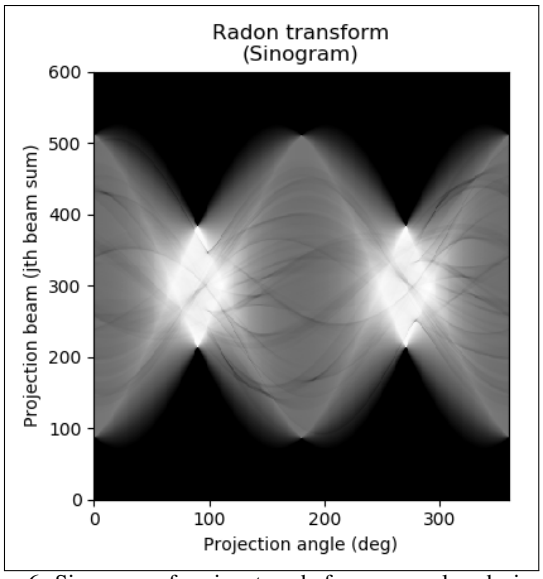

Figure 6. Sinogram of a signature before zero-value decimation and invariance post processing. It has dimensions $k \times \Theta$.

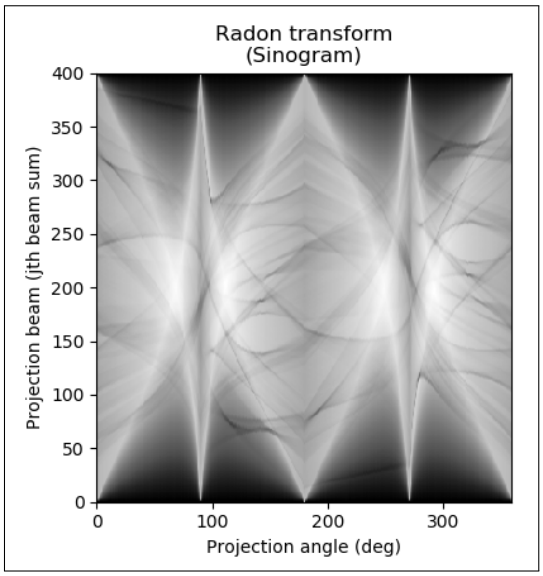

Figure 7. Sinogram of a signature after zero-value decimation and invariance post processing. It has dimensions $d \times \Theta$.

3.3. Feature Extraction

The resulting sinogram from the DRT computed above is still far too large to pass through a SVM classification efficiently. As such, principle component analysis (PCA) is used to extract the P greatest contributing features, where $P < \Theta$. Using PCA, the average DRT feature $\vec{s}_{ave}$ of the sinogram S is computed as:

$$\vec{s}_{ave,i} = \sum_{j=0}^{\Theta} S_{ij}, i = 1, ..., d \tag{2}$$

and is element-wise subtracted each column of S to yield difference matrix S'. The eigenvectors (features) of S' come from the symmetric covariance matrix $(S'S'^T)$ and are found to be the rows of V^T of its singular value decomposition. The $EigenSignatures$ $\vec{e}_i$ can now be computed as the linear combination of each eigenvector with the columns of S' per Ooi et al. [12] as follows:

$$\vec{e_i} = \sum_{j=0}^{\Theta} v_{ij}\vec{s}_j \tag{3}$$

where v_{ij} is the jth value of the ith eigenvector (row of V^T), $\vec{s}_j$ is the jth column of S' and $i = 1, ..., d$.

Finally, the P first $Eigensignatures$ are returned as they correspond to the P highest eigenvalues and so features of the sinogram.

3.4. Discrete Cosine Transform (DCT)

The DCT represents a finite sequence of data points (ex. an image) image as the sum of cosine functions of varying magnitudes and frequencies. It is integral to most image compression techniques due to its beneficial property of clustering the most visually significant information, i.e. lowest frequency coefficients, in the top left corner of a DCT transformed image. Furthermore, the DCT is an orthogonal transformation, meaning the inverse DCT can easily be found and that critical properties (ex. rotation, lengths) of the original image are preserved.

Rather than computing the 2D DCT-II through its formal definition, we exploit its designation as a linearly separable transform and its relation to the discrete Fourier transform (DFT) to make computation faster and simpler.

Since it is linearly separable, we can express the 2D DCT in terms of its corresponding 1D DCT-II transform across first the rows of an image to yield an intermittent matrix A, and then compute the 1D DCT across the columns of A to yield the final resulting matrix B (or vice versa). Per this definition and given an $M \times N$ image img with rows $\vec{img}_i$ we define

$$A = \begin{bmatrix} — & dct1d(\vec{img}_1)^T & — \\ — & ... & — \\ — & dct1d(\vec{img}_M)^T & — \end{bmatrix}$$

and columns $c_{\vec{A},i}$ of A,

$$B = \begin{bmatrix} | & | & | \\ dct1d(c_{\vec{A},0}) & ... & dct1d(c_{\vec{A},N}) \\ | & | & | \end{bmatrix} \tag{4}$$

The 1D DCT is can be found through its relation to the DFT, as the real component of the double length fast Fourier Transform is the DCT with a phase shift in the sinusoidal basis functions. Given vector v, we first create a mirrored

double length vector $\vec{u} = [v_0, v_1, ...v_{N-1}, v_{N-1}, ...v_0]$, as its symmetric property enforces the DCT being real. Then we compute the FFT of $\vec{u}$ to yield x. Finally, we adjust the values of $\vec{x}$ for the basis shift by setting

$$x_j = real\{x_j e^{\frac{-i\pi j}{2N}}\} \tag{5}$$

for j in $1, ..., N$.

The largest coefficients $b_{i,i}$ of B for i in $1, ..., k$ then hold enough energy to represent the bulk of the signature image and are used as the features.

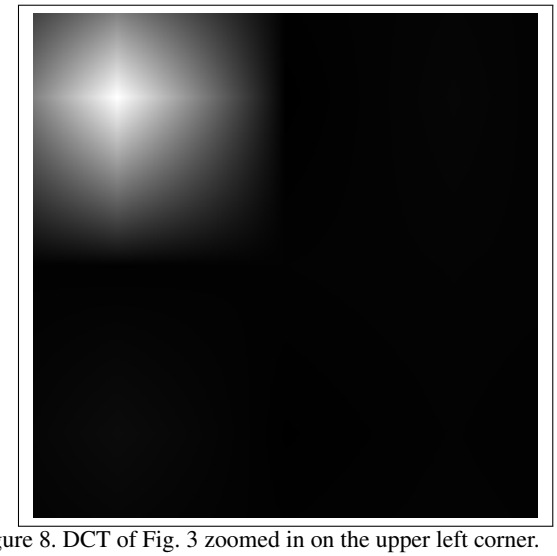

Figure 8. DCT of Fig. 3 zoomed in on the upper left corner.

3.5. Support Vector Machine (SVM)

Once the features were extracted, we used a linear SVM to classify signatures as forgeries or as genuine. Linear SVMs are supervised learning models usually utilized in classification or recognition tasks. They get trained to classify objects into one class or another by fitting a 'line' of best fit in a n-dimensional training set space. It partitions the training data and uses that to make predictions on the test data.

To classify, we made both our test and training sets pairs of input signatures where one is the genuine signature for a user and the other is either also genuine or a forgery. Our classification task is to identify if the second signature is a forgery or not. For this case, we used the difference between the feature vectors for the genuine signature and the questioned signature as the input to the SVM.

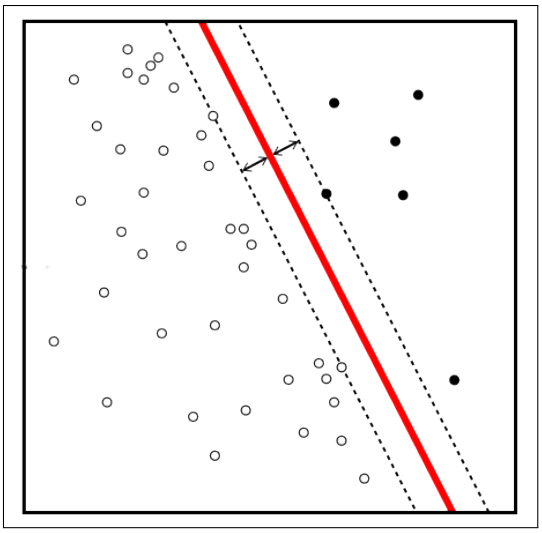

Figure 9. Linear SVM data partitioning [1].

4. Experiments and Results

4.1. Datasets

These results were obtained through testing on two publicly available offline signature databases, namely: the ICFHR 2010 Signature Verification Competition dataset [6] and the Caltech Vision Group Signature Data version 1.0 [8] [9] [10]. The ICFHR La Trobe data set contains 209 images, with 85 genuine signatures written by a single reference writer "A" and 104 skilled forgeries written by 27 people freehand copying the reference signature's characteristics. The test set contains 125 signatures, with 28 valid signatures by a second writer "B" and 97 forgeries, written by 34 forgers per the same specification as the training set. We created data pairs by selecting all possible combinations of genuine-genuine and genuine-forged pairs from each writer giving us 8280 total pairs. We then split the data into a test and training set. We varied the training set size and sampled pairs at random and examined the effects the training set size had on the accuracy.

The Caltech database consists of two sets of subjects, each with genuine signatures and casual forgeries. Set 1 contains 56 subjects, each with 24 genuine signatures and 9 forgeries, while set 2 contains 50 subjects, each with 29 genuine signatures and 9 forgeries. The signatures are saved in ASCII format, i.e. a list of (x, y) coordinates corresponding to the written sample. As such, this data was plotted with the resulting images being saved for later analysis, allowing for us to test our model on many more writers. Our tests still performed well on these images, despite their resolution and accuracy to the original signatures being fairly low. For this dataset, we started using a similar process for selecting samples for the test and training sets. We used five users for our testing, selected at random, and examined the

effects of variable sized training sets on accuracy. This gave us 27550 total pairs. We then increased the number of users to 15. This gave us 250974 pairs allowing us to examine the effects on accuracy as the user pool grew.

4.2. Results and Evaluation

Our first sets of experimentation were to tune the parameters for the feature extractors. For the DRT-PCA method, to ensure that we implemented the process correctly, we tried to match the sinogram images from the paper by Shin Yin Ooi et al. [12] as closely as possible before proceeding. As shown by Fig. 7, we were successfully able to produce a sinogram and perform the zero-value decimation and invariance post processing steps. Once we tuned our parameters, we preceded with the following values for DRT-PCA:

$$k = 600,\ \Theta = 360,\ d = 400$$

We then only take the first principal component for our feature vector (dimension 360×1). We tried to have a similar number of features received from the PCA method for the DCT method. So, for the DCT method we used:

$$k = 18$$

This gives us a 324×1 feature vector for each image. After tuning the parameters, we tested in a fixed size training set of 250 for both datasets. We ran 5 iterations for each method where we randomly selected a training set of 250, trained a classifier, and ran the classifier against the test set. One of our goals when analyzing the data was to minimize the number of false positives (forged signatures that were classified as genuine). So, we examined the recall and precision for both genuine and forged signatures as well to get more fine grained analysis. We noticed that on the same input size, our model did better when trained and tested on fewer users. Additionally, for this input size, the DRT-PCA features outperformed the DCT features. The results we got were as follows:

	DRT-PCA	DCT
Test Accuracy	93%	87%
Genuine Precision	97%	98%
Genuine Recall	94%	87%
Forgery Precision	72%	56%
Forgery Recall	86%	89%
Equal Error Rate	13%	14%

Table 1. Test results for La Trobe dataset (skilled forgeries).

	DRT-PCA	DCT
Test Accuracy	90%	82%
Genuine Precision	78%	60%
Genuine Recall	78%	75%
Forgery Precision	93%	92%
Forgery Recall	94%	84%
Equal Error Rate	18%	22%

Table 2. Test results for Caltech dataset (coarse images of casual and semi-skilled forgeries).

For the La Trobe dataset, we were able to achieve high precision for genuine signatures. In other words, the results showed a low false positive rate. The model did not do as well for genuine precision with the Caltech dataset but it did have high forgery precision, indicating that the model had a low false negative rate. This disparity may be due to the resolution difference between images in the two datasets or the average forgery level present (skilled vs. casual).

We then experimented with the training set sizes, we used the features from each feature extraction technique and we used a parameter r which indicted how much of the training dataset we would use to train the SVM. For the La Trobe dataset (Fig. 10) we can see that the DRT-PCA method is more accurate than DCT. Both methods follow a similar curve for accuracy as the training set sizes increase and eventually flatten out with accuracy above 95%.

Figure 10. Test accuracy for the La Trobe dataset.

For the Caltech database, we found similar results. The dataset for the Caltech database was much larger so we started with a small subset of the users, chosen at random and scaled the number of users included, in the training and test sets. As you can see by Fig. 11 the curves for accuracy mirror those from the previous dataset.

Interestingly, the Caltech dataset results indicate that while the DRT-PCA method works better for smaller training sets, DCT performs slightly better with larger sized training sets.

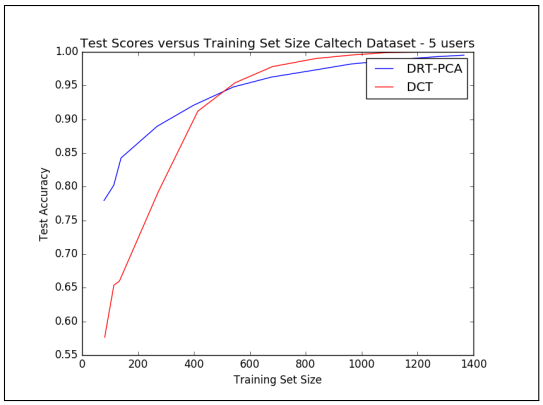

Figure 11. Test accuracy for the Caltech dataset with 5 users.

This indication was contradicted when we increased the dataset size to 15 users. We found lower accuracies using similar training set sizes as before. This shows that the training set size will need to scale larger based off of the number of users and test examples. It also displayed a larger separation between scores for DRT-PCA and DCT. With 15 users, i.e. a dataset size of 250974, DRT-PCA clearly gives better results for the model, as seen below. This is completely surprising, as the DRT-PCA method is more invariant than DCT to global transformations of the signature, i.e. translation, rotation and scaling, which are more likely to occur with a higher number of user signatures.

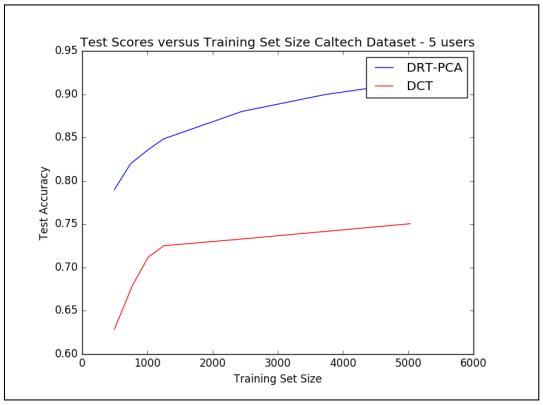

Figure 12. Test accuracy for the Caltech dataset with 15 users.

5. Conclusions

Overall, we were able to create a model which effectively detected forgeries. We were able to examine trade offs in accuracy and efficiency. For our feature extraction algorithms, we observed a trade-off between the DRT-PCA method and the DCT method. While the DRT-PCA method was more accurate overall, it is slower to run than the DCT method. We also noticed a trade-off in the training set size versus accuracy. The lower the training set size, the lower the accuracy. Lastly, we found we needed to scale the training set sizes for the number of users.

For further experimentation, we would explore different methods of classification to see if given our two feature extractors we can achieve better results than using the SVM model. We also would like to further experiment with parameter tuning in the feature extractors to see how the different parameters affect the precision and recall of the model.

The GitHub link for project source code can be found here: `https://github.com/nbutler1/Signature-Detection`

References

[1] Alisneaky. Kernel machines are used to compute a non-linearly separable functions into a higher dimension linearly separable function., 2011. Available at `https://commons.wikimedia.org/wiki/File:Kernel_Machine.png`.

[2] I. Bhattacharya, P. Ghosh, and S. Biswas. Offline signature verification using pixel matching technique. *Procedia Technology*, 10:970 – 977, 2013. First International Conference on Computational Intelligence: Modeling Techniques and Applications (CIMTA) 2013.

[3] S. Chandra and S. Maheskar. Offline signature verification based on geometric feature extraction using artificial neural network, 03 2016.

[4] B. M. H. J. Coetzer and J. A. du Preez. Offline signature verification using the discrete radon transform and a hidden markov model. *Journal on Applied Signal Processing*, 4:559–571, 2004.

[5] S. C. K. Jana, Ranjan Mandal. Offline signature verification for authentication, 09 2015.

[6] L. A. E. v. d. H. Marcus Liwicki, Muhammad Imran Malik and B. Found. Forensic signature verification competition 4nsigcomp2010 detection of simulated and disguised signatures. *Proc. 12th Int. Conference on Frontiers in Handwriting Recognition*, 2010.

[7] R. Mugabe. Signature of robert mugabe clear, 2008. Available at `https://commons.wikimedia.org/wiki/File:Signature_of_Robert_Mugabe_clear.svg#`.

[8] M. Munich and P. Perona. Visual input for pen based computers. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 24(3):313–328, 2002.

[9] M. Munich and P. Perona. Visual identification by signature tracking. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 2003.

[10] M. Munich and P. Perona. Apparatus and method for tracking handwriting from visual input. *US Patent 6,044,165*, filed 6/15/1995, granted 3/28/2000.

[11] E. Özgündüz, T. Sentürk, and M. Elif Karsligil. Off-line signature verification and recognition by support vector machine. 01 2005.

[12] Y. H. P. Shih Yin Ooi, Andrew Beng Jin Teoh and B. Y. Hiew. Image-based handwritten signature verification using hybrid methods of discrete radon transform, principal component analysis and probabilistic neural network. *Applied Soft Computing*, 40:274–282, 2016.