A Comparison of Machine Learning Techniques for Handwritten |Xam Word Recognition

Kyle Williams*
Information Sciences and Technology
Pennsylvania State University
University Park, PA, 16802, USA
kwilliams@psu.edu

Hussein Suleman
Dept. of Computer Science
University of Cape Town
Private Bag X3,
Rondebosch, 7701
South Africa
hussein@cs.uct.ac.za

Jorgina K. do R. Paihama
Dept. of Computer Science
University of Cape Town
Private Bag X3,
Rondebosch, 7701
South Africa
jpaihama@cs.uct.ac.za

ABSTRACT

The Bleek and Lloyd collection contains 19th century handwritten notebooks that document the language and culture of the |Xam-speaking people who lived in Southern Africa. Access to this rich data could be enhanced by transcriptions of the text; however, the complex diacritics used in the notebooks complicate the process of transcription. Machine learning techniques could be used to perform this transcription, but it is not known which techniques would produce the best results. This paper thus reports on a comparison of 3 popular techniques applied to this problem: artificial neural networks (ANN); hidden Markov models (HMM); and support vector machines (SVM). It was found that an SVM-based classifier using histograms of oriented gradients as features resulted in the best word recognition accuracy of 58.4%. Furthermore, it was found that most feature extraction parameters did not have a large effect on recognition accuracy and that the SVM-based recognisers outperform both ANN- and HMM-based recognisers.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Digital Libraries; I.2.6 [**Artificial Intelligence**]: Learning; I.7.5 [**Document and Text Processing**]: Document Capture—*Optical Character Recognition (OCR)*

General Terms

Algorithms, Experimentation, Performance

Keywords

OCR, handwriting recognition, cultural heritage preservation, Bleek and Lloyd Collection

1. INTRODUCTION

The Bleek and Lloyd Collection [22] is a collection of notebooks that document the language and culture of the |Xam and !Kun people who lived in Southern Africa in the late 19th century. The collection has been digitised and digital library systems have been created for preserving this collection and providing access to it via the Web [22, 26]. Furthermore, some tools have been designed for enhanced interaction with the collection [27]. However, these systems are limited in that they mainly operate at the image level and not on the text contained in the notebooks. One way of allowing for access to the text is through the creation of transcriptions by converting the images of the stories into text that can be represented on a computer. Transcriptions allow for a number of ways of interacting with the collection by allowing for the data to be indexed, searched and compared. Transcriptions can also be used in speech-to-text applications and also allow for the text to be reprinted in books. However, manual transcription is a time-consuming and costly process and is generally not a viable option. Fortunately, automatic handwriting recognition has developed sufficiently that it is now viable to manually transcribe a small subset of a collection and then use machine learning techniques to automatically transcribe the remainder of the collection.

While the automatic recognition of handwritten texts is a well-studied problem, automatic recognition of the texts that appear in the Bleek and Lloyd Collection is difficult due to the fact that the script used to represent the characters is complex, not well understood and contains complex diacritics that appear: above characters; below characters; and above and below characters. Furthermore, these diacritics can span multiple characters and can be stacked (see Table 1).

A variety of techniques have been used for handwriting recognition and this paper describes an investigation into which techniques are most appropriate for use on the |Xam script, which is one of the languages that appears in the Bleek and Lloyd Collection. The techniques investigated focus on the use of different machine learning techniques and descriptive features in order to determine which result in the highest recognition accuracies. In doing so, the hope is to identify good techniques for automatically recognising handwritten |Xam texts, as well as other historical texts that are represented in complex scripts. Furthermore, since this discussion forms a basis for future research, some directions for future research based on the findings of this study are discussed.

*Based on work conducted while at University of Cape Town.

SAICSIT '13, October 07-09 2013, East London, South Africa
Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM 978-1-4503-2112-9/13/10...$15.00.
http://dx.doi.org/10.1145/2513456.2513463
.

In doing this, the rest of this paper is structured as follows. Section 2 describes the Bleek and Lloyd Collection while Section 3 presents some related work on the automatic recognition of complex scripts. Section 4 describes the recognition systems that were used as part of this study, while Section 5 presents the experiments that were conducted to investigate the use of the recognisers. Lastly, conclusions are drawn in Section 6.

2. BLEEK AND LLOYD COLLECTION

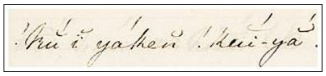
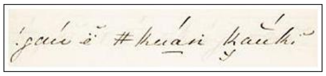
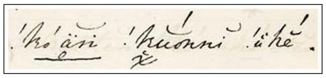
The |Xam speaking people are among South Africa's oldest human inhabitants and it is likely that they have a unique view of the world. However, like the languages of many ancient cultures, the |Xam language has completely died out [22]. As with many ancient cultures, |Xam culture and beliefs were metaphorically encoded and shared in the form of stories. These stories were written down in notebooks by German linguists - Wilhelm Bleek and Lucy Lloyd - in the 1870s and, together with art and dictionaries, have come to be known as the Bleek and Lloyd Collection [22]. In its entirety, the Bleek and Lloyd Collection is a collection of notebooks, art and dictionaries that document the languages and culture of the early human inhabitants of Southern Africa and, more specifically, the languages and culture of the |Xam and !Kun people. The notebooks in the collection contain metaphorical stories and, in most cases, their corresponding English translations appear alongside them. The drawings in the collection complement the stories in the notebooks, while the dictionaries contain English words and their corresponding |Xam or !Kun translations. Figure 1 shows an example of a notebook page, dictionary entry and artwork from the Bleek and Lloyd Collection, respectively.

The Bleek and Lloyd Collection is jointly owned by the University of Cape Town, The National Library of South Africa and the Iziko National Museum of South Africa and is made up of 157 notebooks containing 14128 pages, 752 drawings and over 14000 dictionary entries [22]. In 1997 the collection was recognised by UNESCO as a Memory of the World Collection [22], thereby illustrating the importance of the collection, and in 2003 the Lucy Lloyd Archive and Research Centre at the Michaelis School of Fine Arts undertook to digitally preserve the collection.

2.1 |Xam Diacritics

The complexity in automatically transcribing the texts that appear in the Bleek and Lloyd Collection arises because of the script used to represent the text. The script is complex due to the diacritics that appear above characters, below characters and both above and below characters. Diacritics can also span multiple characters. Thus far, more than 64 different combinations of single and stacked diacritics have been discovered[1] that appear above characters, 13 that appear below characters and 60 that appear both above and below characters. Table 1 shows some of the types of diacritics that appear in the texts, starting with simple diacritics that appear above characters, then diacritics that appear above and below characters and then diacritics that are stacked and that span multiple characters. One of the complexities introduced by the diacritics is that some characters may only differ by their diacritics, while their Latin base characters remain the same. This is shown in Figure 2, where several variations of an *a*-character are shown.

Table 1: Images of text with diacritics from the notebooks and their transcriptions

Image	Transcription
	!kŭ̍ ï̆ ya̍ken̆ !kui̍̆-yă̍ .
	!gau̍ ë̆ #kua̱̍n $\underset{y}{\text{k}}$au̍̆kĭ
	!ko̍ḁ̱̈̆ṅ $\underset{\overset{c}{x}}{\text{k}}$ ŭ̍onnĭ !ŭhĕ̍ .

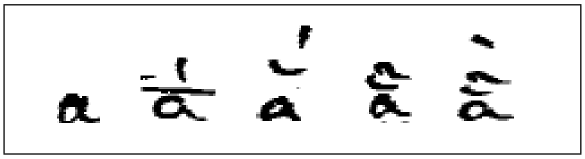


Figure 2: Example of variants of a |Xam character

A key requirement for automatic transcription is the ability to handle the complexities introduced by these diacritics. In this study, a baseline analysis of machine learning techniques and features is applied in order to determine how well they are able to handle these complexities.

3. RELATED WORK

The automatic recognition of relatively simple, well-understood scripts is a well-studied problem with state of the art recognition systems achieving recognition accuracies of around 80% [8]. However, for more complex scripts, the recognition accuracies are often much lower. For instance, a system designed for recognising Chinese handwriting was only able to achieve a recognition accuracy of 55.58%, which was an improvement on the accuracy of a popular commercial handwriting recognition system [21]. Complex scripts exist all over the world, especially in the East. For instance, the Kannada Kagunita script is formed by combining 34 consonants with 15 vowels in order to create 510 uniquely shaped *Aksharas* [19]. An approach was taken to recognising these Aksharas where the regions of consonants and vowels in an Akshara were automatically detected, separated and then recognised independently, thus reducing the number of symbols that need to be classified to 49. Using this approach, vowels were recognised with 85% accuracy and consonants with 59% accuracy [19]. An approach like this is possible since the two components that make up an Akshara are well-defined; however, for the |Xam script, this is not the case, thus rendering an approach like this infeasible.

In a study on the recognition of Vietnamese script, it was argued that segmenting diacritics from base characters is a difficult task [16]. Thus, a 3-layer classifier system was used where the first layer grouped symbols by their base characters and the second and third layers distinguished among the diacritics [16] to achieve recognition accuracies of around 95% for 95 characters. However, this was an online recognition task whereas the recognition of the |Xam script is an offline task. The online recognition of handwriting is usually considered as being easier than offline recognition since additional information, such as temporal information, can be exploited.

[1] New diacritics are constantly being discovered as the text is explored.

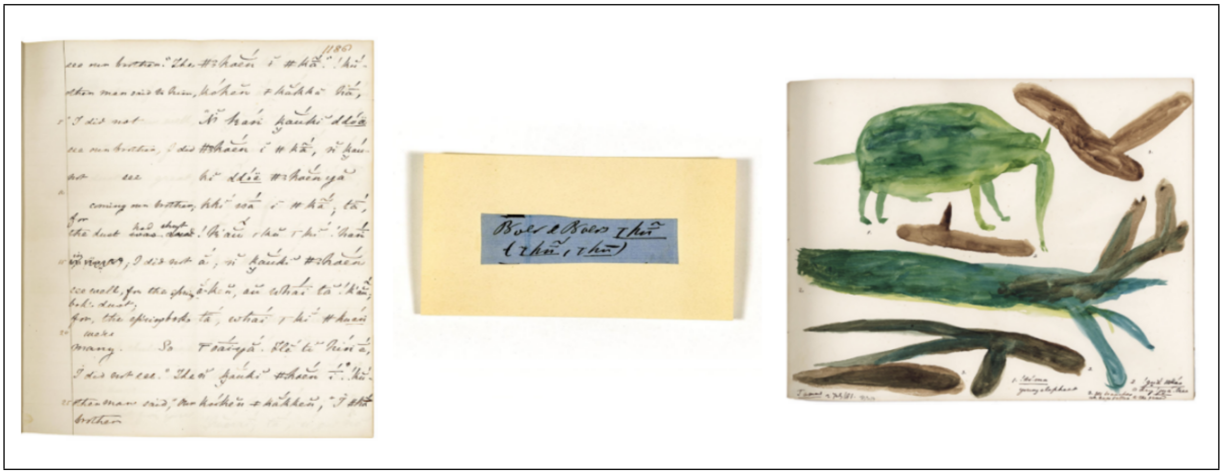


Figure 1: A notebook page, dictionary entry and artwork from the Bleek and Lloyd Collection

Arabic texts, like the |Xam texts, contain complex diacritics. In one approach to recognising Arabic text, 34 of the most commonly occurring Arabic sub-words were recognised using a support vector machine without paying specific attention to the diacritics [5] and a recognition accuracy of 84.3% was achieved. This is a relatively high recognition rate, though the number of classes that needed to be recognised (34) was relatively low. In a similar study, whole word recognition was applied to 500 Tunisian town/village names using an artificial neural network and a recognition accuracy of 82.5% was achieved [3]. In another study involving Arabic character recognition, in addition to moment features, the "number of dots" and "number of holes" were also extracted, which described the diacritics in more detail [1]. Haboubi et al. [10] also presented an evaluation of four different features for handwritten Arabic recognition and found that orientation-based features performed poorly for Arabic texts.

As this section has shown, a number of approaches have been taken to try to recognise complex scripts, such as: not paying special attention to the complexities of the script; separating the script into its constituent parts and recognising them separately; using a multilayer classifier; and using features specifically designed to address the complexities of the script. What this section has shown is that the recognition of complex script often requires the creation of systems that have been designed specifically to deal with the complexities of the script.

4. RECOGNITION SYSTEMS

A series of recognisers were built to investigate different techniques for handwritten |Xam word recognition. The recognisers made use of different descriptive features and machine learning algorithms. For word recognition, rather than recognising the individual symbols that make up a word, each word was recognised as a whole and thus represented a single pattern that needed to be classified. The design of the word recognisers is described below.

4.1 Machine Learning Techniques

The machine learning techniques that were used for the recognisers were: Support Vector Machines, Artificial Neural Networks and Hidden Markov Models. These machine learning techniques were chosen since they have previously been used in a number of studies for handwriting recognition and are popular and well-known learning approaches. The implementation and design of these machine learning techniques is briefly described below.

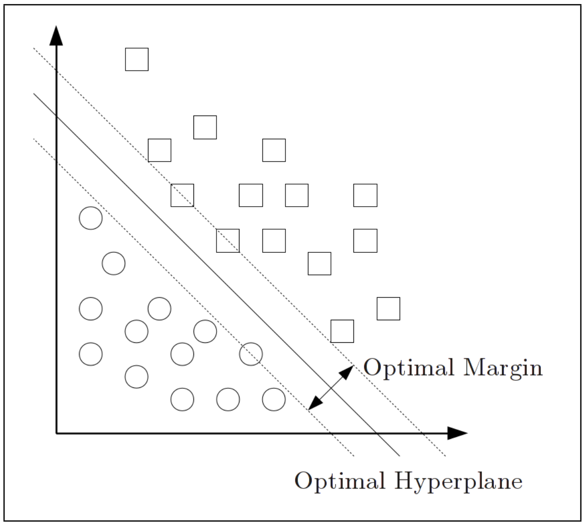


Figure 3: Optimal hyperplane that maximises the space between classes

4.1.1 SVM-based Recogniser

A Support Vector Machine (SVM) is a machine learning technique for classification where the goal is to find hyperplanes capable of separating points in hyperspace [6] and that maximises the space between classes. Figure 3 shows an example of a SVM for a 2-class classification problem. SVMs have been used in a number of studies for handwriting recognition [1, 16] and in this study they are used for recognizing |Xam words. The SVM used in this study makes use of a Radial Basis Function (RBF) kernel, which is defined by two parameters - C and λ. The values for these parameters were found through a grid search where pairs of values for C and λ were tested for the training set in order to see which pair produced the best results [12]. The classification approach used was one-vs-one classification where $\frac{N(N-1)}{N}$ 2-class support vector machines were constructed for each pair of different classes.

4.1.2 ANN-based Recogniser

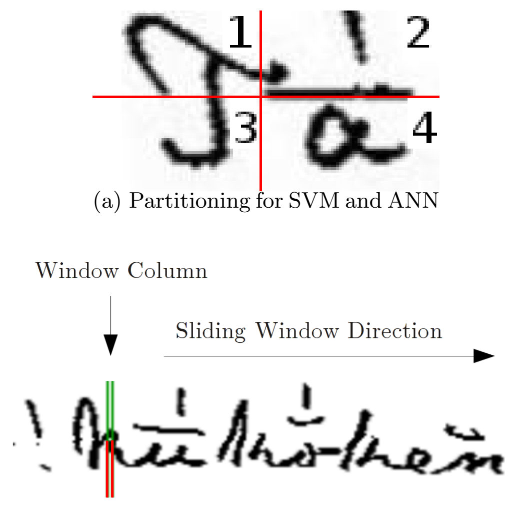
Artificial Neural Networks (ANNs) attempt to mimic the biological neural networks that exist in the human brain for very specific tasks and the main information processing unit in a neural network is the neuron. In this study, a multi-layer neural network is used for |Xam word recognition. The architecture of the network involves an input layer that represents a set of features, a hidden layer, and an output layer that represents the output classes. The size of the input and output layers was determined by the number of features for each training sample and the number of word classes that needed to be classified, respectively. The size N of the hidden layer was a function of the size of the input and output layers, where $N = \{S_I, \frac{S_I}{2}, S_O, \frac{S_O}{2}, \frac{S_I+S_O}{2}\}$, where S_I is the size of the input layer and S_O is the size of the output layer. ANNs were created for each value N and the results reported are for the best performing architecture. The ANN-based recognisers all made use of the iRprop training algorithm [14] and the *tanh* activation and error functions. Ensembles of 10 ANNs were created for each experiment in order to minimise the effect of random weights during ANN initialisation [20]. The ANNs in the ensemble differed by their random initialisation weights and were combined by averaging their outputs [20].

4.1.3 HMM-based Recogniser

A Hidden Markov Model (HMM) is a Markov model in which the states of the model are not directly observable, but rather are observed through another stochastic process [18]. This is in contrast to regular Markov models in which states translate to observable physical events. In this study, the HMM-based recogniser that was used for handwriting recognition was based on a continuous left-to-right HMM, which is defined by two parameters: the number of emitting states and the number of Gaussians. These were set to 12 and 16 since they were empirically found to perform well for the |Xam texts using a technique for estimating good values for these parameters [9]. The HTK library [29], which makes use of modified versions of the Baum-Welch and Viterbi algorithms, was used for HMM-based recognition. As previously mentioned, the recognisers recognise each word as a single pattern and thus HMMs were built for each word and the output of the recogniser was constrained to a single word.

4.2 Features

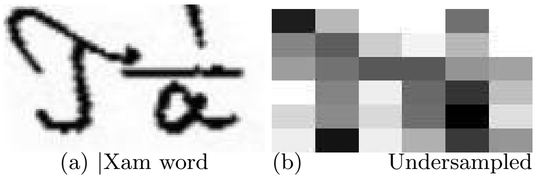
The machine learning techniques described above were paired with different descriptive features in order to determine which are most appropriate for |Xam handwriting recognition. It has been speculated that there are hundreds of different features that can be used for handwriting recognition [4], making implementing and evaluating every possible feature set an impossible task. Therefore this study has instead focused on a small subset of features used in the literature. A total of 6 different features were used, all of which have been used in various studies for the automatic recognition of handwritten texts and that have varying levels of complexity. The features describe different properties of the images, such as the distribution of pixels, the directions of strokes and the statistical properties of the image. Therefore, while not a complete evaluation of all possible features, this study still provides a valuable comparison of different types of features. The generic method for feature extraction and a description of each feature used in this study is described below.



(a) Partitioning for SVM and ANN

(b) Sliding window and partitioning for HMM

Figure 4: Partitioning of words into cells for feature extraction



(a) |Xam word

(b) Undersampled bitmaps

Figure 5: Undersampled bitmap features

4.2.1 Feature Extraction

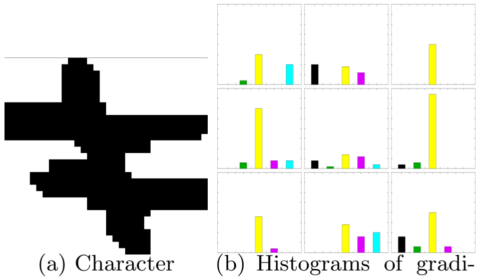
For SVM and ANN-based recognition, features were extracted from the whole word images, which were also partitioned into cells of various sizes so as to investigate the effect of varying degrees of local and global descriptors. Features were extracted from each cell, thereby creating a feature vector that was a concatenation of the features from each cell. For HMM-based recognition, features were extracted using a left-to-right sliding window, which, like SVM and ANN-based recognition, was divided into cells, except only in the horizontal dimension. Figure 4 shows an example of how features were extracted from word images. The features used in this study are briefly described below.

4.2.2 Undersampled Bitmaps

Undersampled bitmaps describe the distribution of pixels in the Cartesian plane. They are calculated by dividing an image into cells and then, for each cell, counting the number of foreground pixels and normalising it over the range [0-1] [17]. The extraction of these features varied in the number of cells that the image or sliding window (for HMMs) was divided into. Figure 5 shows an example of undersampled bitmap features for a |Xam word.

4.2.3 Marti & Bunke Features

Marti and Bunke [15] proposed nine geometric features ($F_1 - F_9$) that are extracted using a sliding window. The



(a) Character (b) Histograms of gradients for cells in image

Figure 6: HoG features

first three features are the weight of the window, the centre of gravity of the window and the second order moment of the window. Features 4 and 5 are the lower and upper positions of the contours of the window. Features 6 and 7 are the gradients of the upper and lower contours of the window. Feature 8 is the number of black to white transitions and feature 9 is the number of black pixels between the lower and upper contours of the window [15]. These features are extracted in the same way for all images.

4.2.4 Geometric Moments

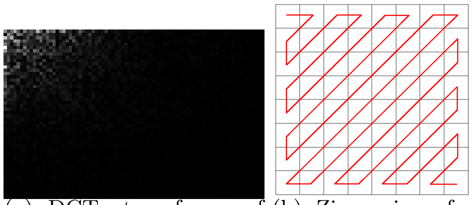
The first four of Hu's geometric moments [13] were used as features in different combinations. The first two moments (M_1, M_2) describe the variance of an image and the third and fourth moments (M_3, M_4) describe the skew of an image. These moments are fully invariant to rotation, translation and scale [13]. The extraction of these features varied in terms of the number of cells that the image or sliding window was partitioned into as well as which of the first four of Hu's geometric moments were used as features.

4.2.5 Histograms of Oriented Gradients

Histograms of Oriented Gradients (HoGs) are a set of descriptors that were first introduced for detecting humans in images [7]. The basic thought behind HoGs is that local object shape and appearance in an image can be characterised by the distribution of local gradients. This distribution of local gradients is found by dividing an image into a series of equally-sized cells and then, for each cell, compiling a list of the gradients of each pixel in the cell and creating a histogram of gradients for that cell. Figure 6 shows an example of a character that has been partitioned into 9 cells and the HoGs for each cell.

To find the distribution of local gradients, an image is first divided into a series of equally-sized cells. Then, for each pixel in each cell, the orientation θ is calculated as $\theta = \arctan\frac{I_x}{I_y}$, where I_x and I_y are the x and y locations of the pixel. The orientation is then transformed into a gradient as $\alpha = \theta \times \frac{180}{\pi}$. Each pixel casts a weighted vote for one of 9 orientation-based histogram channels with weight $W = \sqrt{I_x^2 + I_Y^2}$. The histograms can be grouped into descriptor blocks and normalised [7]; however, this was found to have a negligible effect in this study.

The way in which features were extracted in this study was the same as that used by Howe et al [11]. In this approach, features are extracted using different cell sizes, which are referred to as *resolutions* and the features from each resolution are concatenated to create a single feature vector. In this study, the extraction of these features varied in terms of the size of the resolutions used and up to three resolutions were used at a time.



(a) DCT transform of word in Figure 5 (a) (b) Zigzagging for DCT coefficient extraction

Figure 7: DCT transform showing most of the variance in the low frequency components

4.2.6 Gabor Filters

A Gabor filter is a Gaussian function modulated by a complex sinusoid in both the spatial domain and frequency domain and contains both a real and imaginary part [5]. Gabor filters are orientation specific [5] and thus they have been used for extracting stroke information from characters for use as features for character recognition [24]. Basically, Gabor filters are applied to an image at different orientations to extract orientation information.

The approach used to create Gabor filter-based features is the same as that of Chen at al [5] and is briefly described here. For a frequency f, which is based on the width of a handwriting stroke, and a stroke orientation angle θ, a Gabor filter is applied to an image. In order to emphasize salient pixels in the response of the Gabor filter, the mean magnitude - M_{mean} - of the response is computed and pixels whose magnitude exceeds M_{mean} are considered salient. The total number of salient pixels - S_T - is recorded and then, to calculate features, the image or sliding window is split into cells. For each cell, the number of salient pixels in that cell - S_C - is computed and the feature for that cell is then given by $\frac{S_C}{S_T}$. The features for all cells are concatenated to create a single feature vector. In this study, the extraction of Gabor filter-based features varied in terms of the size of cells used for salient feature calculation and Gabor filters are applied for each combination of $f = \cos\frac{1}{\lambda}, \lambda = \{2, 4\}$ and $\theta = \{0, \frac{\pi}{4}, \frac{\pi}{2}, \frac{3\pi}{4}\}$. These values were based on previous studies where they were found to work well [5, 24].

4.2.7 Discrete Cosine Transform

The Discrete Cosine Transform (DCT) expresses a function or signal in terms of a sum of different cosine functions at different frequencies [16]. There are eight kinds of DCT, though the Type-II DCT is the most common and is the one used in this research. Low frequency regions of the DCT contain most of the energy from the transform and encode most of the variance of an image [2] and thus can be used as features, while the high frequency components can be discarded [2]. Figure 7 (a) shows the DCT transform of Figure 5 (a), which shows most of the variance in the low frequency components.

The 2-D coefficients of the DCT are converted to a 1-D vector using a technique called zig-zagging [3] (Figure 7 (b)), the purpose of which is to extract the low frequency coefficients from the DCT first. In this study, the extraction of features varied in terms of the size of the cells that the image or sliding window was partitioned into for the DCT-II calculation and the number of DCT coefficients used as features.

5. EXPERIMENTS

5.1 Experimental Methodology

Each machine learning technique and feature combination was evaluated in terms of its performance with varied parameters for the features. Due to a limited amount of training and testing data, 10-fold cross validation was used for evaluation. In 10-fold cross validation, the data is randomly partitioned into 10 folds and 9 folds are used for training while the other is used for testing. The folds are then rotated and the final result reported is the average of all folds. The data was randomly partitioned into folds that were kept constant for all experiments.

5.1.1 |Xam Word Corpus

A corpus of handwritten |Xam words and their transcriptions was created for use as a dataset in this study [25]. The words in the corpus were binarised and normalised to 64x64 pixel images as part of a preprocessing step [25]. For word recognition, the task was to recognise each word in its entirety as a single pattern. Thus, in order to ensure that there were sufficient samples per class, the corpus was sampled so that only words that had more than 5 samples were included for use in this study. Furthermore, each word in this test dataset was checked to ensure that it was correctly labelled and equivalent labels were merged. The total number of words in the test data set was 1248 and the number of word classes to be recognised was 81. If diacritics are ignored i.e., if it is imagined that they are not there, then there are 39 different word classes; thus, it can be concluded that approximately half of the word classes differ from others only by their diacritics. The descriptive statistics of the data set in terms of the number of samples per class are as follows: the minimum was 5; the maximum was 110; and the mode, median, and mean were 5, 9, and 15.41 respectively.

5.1.2 Feature Extraction

When extracting features, each image or sliding window was partitioned into cells along each dimension (though only horizontal for the HMM-based recognisers) and then the cells were used for feature extraction as described in Section 4.2. Images were partitioned into the following number of cells along each appropriate dimensions: 1 (i.e., no partitioning), 2, 4 or 8. The only exception was Marti & Bunke features, where no partitioning took place since the features characterize the whole sliding window. For HMM-based recognition, the width of the sliding window for undersampled bitmaps, Marti & Bunke features, geometric moments and DCT features was 1 pixel, while for histograms of oriented gradients and Gabor filter-based features it was 4 pixels and there was a 3 pixel overlap between successive windows.

In addition to partitioning the image and sliding windows into cells, some features had additional extraction parameters. Different combinations of Hu's geometric moments were experimented with by extracting and appending new moments on to the feature vector. The following combinations of Hu's first four geometric moments - M_{1-4} - were used: M_1, $M_{1,2}$, $M_{1,2,3}$, $M_{1,2,3,4}$, i.e. experiments were conducted with only the first moment, then the first and the second, etc. For histograms of oriented gradients, the sizes of the three resolutions - R_1, R_2, R_3 - at which features are extracted were: [64,32,16], [32,16,8], [16,8,4], [64,32,0], [32,16,0], [16,8,0], [64,0,0], [32,0,0], [16,0,0]. For each triple, features were extracted at each resolution and combined to create a single feature vector. For SVM and ANN recognisers, the cell sizes are $R \times R$ and for HMM recognisers the cell sizes are $R \times W$, where W is the width of the sliding window. Lastly, the number of DCT coefficients - C - was varied such that $C = \{1, 10, 20\}$.

5.2 Machine Learning Techniques and Different Features

In this section the results of the experiments regarding the use of machine learning techniques with different features are discussed from two perspectives: first from the perspective of the individual features in terms of their parameters for feature extraction; and then the performance of the different recognisers is discussed when the best parameters for feature extraction are used.

5.2.1 Parameters for Feature Extraction

Table 2 shows the recognition accuracies for the different recognisers with different feature extraction parameters. Where values are missing for the cell size of 64 for undersampled bitmaps and Gabor filters, it is due to the fact that the word images needed to be divided into at least two cells for feature calculation. Similarly, for the DCT features, some of the HMM and ANN models became too large when the cell sizes became too small and thus models were not trained in these cases.

Table 2 shows that the size of the cells that an image or sliding window is partitioned into seems to be correlated with recognition accuracy, with smaller cell sizes (and thus more partitions and more descriptors) generally leading to higher recognition accuracies for all but one feature. For instance, for undersampled bitmaps, Hu's geometric moments and Gabor filters, the recognition accuracies are highest for the smaller cell sizes, though this is not the case for DCT features. These findings suggest that extracting features from local areas leads to better recognition accuracies for the |Xam texts. One possible reason for this being the case is that many smaller regions better capture the contribution of the diacritics than fewer larger ones.

There are also a number of observations that can be made about specific features. For instance, Table 2 shows that, in most cases, the use of additional moments beyond the first led to a decrease in recognition accuracy, suggesting that the first moment had more descriptive power by itself than when it was combined with other moments. For the histograms of oriented gradients, there is no evidence to suggest that certain resolutions will necessarily result in higher recognition accuracies, though when there are very few descriptors, as is the case for when the three resolutions are [64, 0, 0], the performance is relatively poor. Furthermore, for the HMM-based recogniser, the histograms of oriented gradients performed extremely poorly. Various feature extraction parameters were experimented with and consistently led to the same results. It was speculated that the reason for the poor HMM performance is that the relatively thin width of a sliding window column does not result in histograms of oriented gradients that are as descriptive as those when larger cells are used; however, when wider sliding window columns were used, no improvement in performance occurred. Lastly, for the DCT features, for large cell sizes, the use of more coefficients seems to lead to higher recognition accuracy; however, as the cell sizes decrease and the number of descriptors used as features thus increases, the number of coefficients does not appear to have as significant an effect on the recognition accuracy. The results seem to suggest that the number of descriptors that are used could be more

Table 2: Recognition accuracies for different recognisers with different feature extraction parameters

Undersampled Bitmaps				
Cell Size	*64*	*32*	*16*	*8*
SVM	-	24.03	43.11	50.73
ANN	-	22.51	43.75	46.71
HMM	-	30.37	32.13	35.02
Marti & Bunke				
SVM		47.23		
ANN		40.30		
HMM		44.47		
Hu's Geometric Moments				
Cell Size	*64*	*32*	*16*	*8*
SVM	$M_1 = 9.53$ $M_{12} = 10.01$ $M_{123} = 10.26$ $M_{1234} = 12.01$	$M_1 = 10.17$ $M_{12} = 9.45$ $M_{123} = 8.57$ $M_{1234} = 7.85$	$M_1 = 21.15$ $M_{12} = 21.71$ $M_{123} = 20.50$ $M_{1234} = 18.98$	$M_1 = 27.96$ $M_{12} = 25.24$ $M_{123} = 24.43$ $M_{1234} = 24.59$
ANN	$M_1 = 8.94$ $M_{12} = 10.57$ $M_{123} = 12.20$ $M_{1234} = 11.38$	$M_1 = 13.82$ $M_{12} = 9.76$ $M_{123} = 11.38$ $M_{1234} = 11.38$	$M_1 = 18.82$ $M_{12} = 9.53$ $M_{123} = 9.05$ $M_{1234} = 9.37$	$M_1 = 23.79$ $M_{12} = 9.94$ $M_{123} = 8.97$ $M_{1234} = 8.97$
HMM	$M_1 = 0$ $M_{12} = 4.64$ $M_{123} = 3.76$ $M_{1234} = 3.92$	$M_1 = 23.80$ $M_{12} = 22.44$ $M_{123} = 22.28$ $M_{1234} = 21.15$	$M_1 = 26.05$ $M_{12} = 24.76$ $M_{123} = 22.51$ $M_{1234} = 22.2$	$M_1 = 27.16$ $M_{12} = 24.76$ $M_{123} = 24.03$ $M_{1234} = 23.24$
Histograms of Oriented Gradients				
R_1, R_2, R_3	*[64,32,16]*	*[32,16,8]*	*[16,8,4]*	
SVM	58.24	58.49	51.92	
ANN	36.58	35.71	-	
HMM	1.28	0.56	1.04	
R_1, R_2, R_3	*[64,32,0]*	*[32,16,0]*	*[16,8,0]*	
SVM	49.75	57.92	57.45	
ANN	35.06	43.40	47.60	
HMM	1.04	1.04	0.48	
R_1, R_2, R_3	*[64,0,0]*	*[32,0,0]*	*[16,0,0]*	
SVM	22.51	48.71	57.76	
ANN	18.01	38.11	51.76	
HMM	0.64	1.2	1.12	
Gabor Filters				
Cell Size	*64*	*32*	*16*	*8*
SVM	-	38.46	47.03	53.21
ANN	-	31.63	40.40	46.63
HMM	-	37.26	36.71	36.78
Discrete Cosine Transform				
Cell Size	*64*	*32*	*16*	*8*
SVM	$S_1 = 12.74$ $S_{10} = 41.27$ $S_{20} = 49.84$	$S_1 = 21.15$ $S_{10} = 49.53$ $S_{20} = 49.52$	$S_1 = 44.64$ $S_{10} = 47.92$ $S_{20} = 43.66$	$S_1 = 50.16$ $S_{10} = 44.07$ $S_{20} = 39.02$
ANN	$S_1 = 12.58$ $S_{10} = 22.98$ $S_{20} = 31.56$	$S_1 = 22.27$ $S_{10} = 37.49$ $S_{20} = 40.62$	$S_1 = 43.03$ $S_{10} = 48.80$ $S_{20} = 47.99$	$S_1 = 48.96$ $S_{10} = 48.72$ $-$
HMM	$S_1 = 2.17$ $S_{10} = 38.47$ $S_{20} = 33.73$	$S_1 = 43.35$ $S_{10} = 34.61$ $S_{20} = 30.76$	$S_1 = 39.5$ $S_{10} = 31.41$ $-$	$S_1 = 38.86$ $-$ $-$

important than the DCT frequency of the coefficients used as features; however, to confirm this it would need to be tested with lower frequency descriptors as features.

5.2.2 Performance of Recognisers

Figure 8 shows the recognition for each machine learning technique/feature combination using the best performing feature extraction parameters.

As can be seen from Figure 8, the highest word recognition accuracy of 58.49% was achieved when an SVM was paired with histograms of oriented gradients and the poorest word recognition accuracy was achieved when histograms of oriented gradients were paired with a HMM. Overall, for word recognition, the SVM recognisers consistently outperformed the others, regardless of the features that were used. The ANN word recognisers were the second best performing word recognisers overall and followed the same trend as the SVM recognisers for all

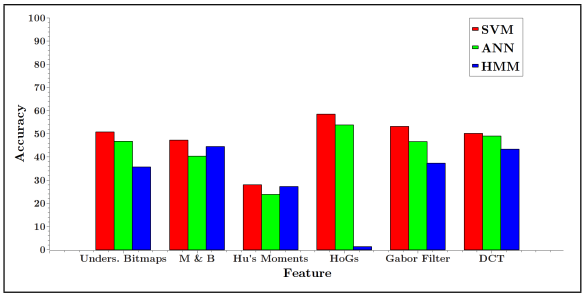


Figure 8: A comparison of the best performing feature parameters for each machine learning technique

features, achieving a maximum recognition accuracy of 53.76%. However, for some features, the HMM word recognisers performed better than the ANN recognisers. Overall, however, the HMM word recognisers performed the worst, achieving a maximum word recognition accuracy of 44.47% when Marti & Bunke features were used.

Figure 8 reveals a number of things concerning the use of different machine learning techniques, different descriptive features and the relationships among them. Firstly, the findings suggest that, in some cases, performance is dependent on the right combination of machine learning technique and descriptive feature. For instance, the use of histograms of oriented gradients as features led to the best performance for SVM and ANN-based recognition and the worst performance for HMM-based recognition. The exact reason for the poor HMM-based recogniser accuracy is unclear as various feature extraction parameters were experimented with and consistently led to the same results. Similarly, the figure reveals that, for HMM-based recognition, the Marti & Bunke features resulted in the best performance, while, on average, they led to the second worst performance for SVM and ANN-based recognition. Figure 8 also reveals that some features, such as undersampled bitmaps, Hu's geometric moments, Gabor filter-based features and DCT coefficients, ranked equally for the different machine learning techniques. However, the best performing features were in fact the ones whose performance varied the most for the different machine learning techniques and thus, when choosing a feature for handwriting recognition, the choice should be made while considering the machine learning technique to be used.

The choice of machine learning technique depends on a number of factors, such as the type of data that is to be recognised. For instance, though it was not done here, it may be possible to segment words into their individual characters and recognise the characters individually; however, when it is not suitable to do so then a HMM or some other machine learning technique that does not require segmentation may be appropriate. Furthermore, the automatic segmentation that occurs as part of the recognition process when using a HMM may make it a better segmenter than other approaches that occur as a preprocessing step. Similarly, if units such as characters or whole words are being recognised then a SVM or ANN may be more appropriate for |Xam script recognition.

5.3 Hybrid Features

A second experiment was conducted where the three best features for each machine learning technique were combined to see what effect (if any) it would have on performance. To create the hybrid features set, the individual feature vectors were concatenated for each machine learning technique. Table 3 shows the recognition accuracies using hybrid features, the best performing single feature and the difference between them (in percentages).

Table 3: Performance of individual features compared to hybrid features for all machine learning technique

MLA	Best individual	Best Hybrid	Change
SVM	58.49%	57.77%	-1.23%
ANN	53.76%	53.60%	-0.29%
HMM	44.47%	44.79%	0.72%

Table 3 shows that, in all cases except HMM-based word recognition, where the improvement was very low, the use of hybrid features had a negative effect on the performance of the recognisers. This goes against intuition, which suggests that hybrid features should improve the performance of a recogniser rather than decrease performance. A possible reason for the decline in accuracy could be that the combined features actually introduced increased variation within classes through multiple descriptors. Another possible explanation could be that the three best features all summarised the same information and thus combining them did not actually add any additional descriptive power. Another possible reason, still, is that the increase in descriptors from hybrid features resulted in more parameters that needed to be trained for the machine learning techniques and, thus, the parameters could have been over-tuned to fit the training data.

While these findings suggest that hybrid features do not offer any improvement on the best performing individual features, care should be taken not to generalise the findings. There are several reasons for this. Firstly, during experimentation it was found that hybrid features did improve the performance of some of the individual features, though not the best performing individual features, thereby suggesting that hybrid features can in fact improve the performance of a recogniser, though that did not occur in this study. Secondly, the hybrid features used in this study only represent a small subset of the potentially infinite number of hybrid features that can be created by combining the previously mentioned hundreds of features used for handwriting recognition. Lastly, equal weightings were used when combining the features and it is possible that weighting the features differently could have an effect on recognition accuracy, for instance, by having some features provide the weighted majority of the information and others only being supplementary. Thus these findings should be taken in context. In this study, hybrid features, in general, offered no improvement over the best performing individual features and, in most cases, led to a decrease in performance. However, it is possible that under different circumstances they may lead to an improvement.

5.4 Discussion

These baseline experiments have shown that the recognition of |Xam texts is a difficult task. Thus, the question arises as to how recognizers can be constructed that can achieve higher recognition accuracies and that are applicable not only to the |Xam texts, but also to other complex scripts, such as the !Kun script in the Bleek and Lloyd Collection. Since the complexities in recognizing the |Xam scripts arises as a result of the diacritics, special attention should be paid to diacritics during the design of future recognition systems. There are two obvious options for doing this: the diacritics could be segmented from the base characters and recognized separately before being re-attached; however, segmentation of diacritics is considered a difficult task [16]. An alternative is to design a multi-layer classifier that ignores diacritics during the first level of classification, assigns characters to classes based on their base characters and then recognizes the variants of characters with the same base character at a second level in the classifier. A similar approach to this was conducted for the |Xam texts and found to improve recognition accuracy by about 10% [28]. However, the difficulty in this approach is that the individual characters need to be segmented to make this approach viable. Alternatively, machine learning techniques that are able to perform automatic segmentation through optimal alignment, such as HMMs, could be used to solve this problem. Indeed, this is likely to be a good starting point for investigating new methods for recognizing |Xam script and other similar scripts and the findings regarding the use of features with HMMs presented in this study could be a good starting point. Similarly, if the individual characters are segmented, the findings from this study suggest that a SVM- or ANN-based recognizer may perform well to recognize the individual characters.

6. CONCLUSIONS

The automatic recognition of the texts that appear in the Bleek and Lloyd Collection is difficult due to the complex diacritics that are used in the |Xam script. In this study, a number of techniques for word recognition were investigated in order to gain insight into which approaches are most suitable for these, and perhaps similar, texts. The highest recognition accuracy was achieved when a SVM was used with histograms of oriented gradients. It was found that the feature extraction parameters for the features used in this study did not seem to have a significant effect on the recognition accuracy for the |Xam texts; though partitioning images into smaller cells and extracting local features did lead to an improvement in accuracy. Furthermore, in comparing machine learning techniques, the SVMs performed better than the ANNs, and the ANNs better than the HMMs and some descriptive features only worked well with certain machine learning techniques when recognizing these texts.

As can be seen from the accuracies achieved, the automatic recognition of |Xam script is a complex task and this complexity can be attributed to two main factors: the complexities of the script; and the relatively small size of the data set used. However, this does not take away from the main contribution of this paper, but rather suggests that additional research into the recognition of these texts will be worthwhile. It is hoped that these findings can be used not only for the design of future recognisers for the |Xam script but also for other similar scripts.

While automatic handwriting recognition systems have successfully been used for well-known and well-understood scripts with relatively small character sets, perhaps they cannot be applied as successfully to more complex scripts. For instance, large collections of documents are likely to exist for well-known and well-understood scripts, thereby allowing for large amounts of training data; whereas more complex scripts are more likely to belong to smaller collections. Furthermore, data that is relatively uniform is needed in order to create robust recognition systems. This was not the case for the |Xam texts and may not be the case for other complex scripts. Thus it could be argued that, given the complexities of the |Xam script and other scripts, it is not worth creating automatic handwriting recognition systems since the accuracies that can be achieved are perhaps too low to be very useful or the complexities in designing the system and investigating techniques are not worth the effort. Thus, for smaller collections, it may be more appropriate to simply manually transcribe the collections or make use of collaborative computer-human transcription techniques, which have been shown to reduce the effort required to correct transcription errors [23].

Finally, the importance of transcriptions of historical documents should not be underestimated. It has already been discussed in the introduction how transcriptions can be used to improve digital library systems by allowing for the provision of enhanced services. However, more importantly, transcriptions of historical documents serve as a means of preserving the documents' contents and ensuring that, regardless of what may happen to the physical artefacts, the knowledge and information that they contain will remain available and accessible to all. This study has taken us one step closer to achieving that goal.

6.1 Future Work

Since this study acts as a baseline for future investigation into recognizing |Xam and other equally complex scripts, there are many possibilities for future work that are worth discussing. For instance, future work could investigate the use of additional machine learning techniques and descriptive features for |Xam handwriting recognition. However, this study has shown why the automatic recognition of |Xam texts is difficult and thus future work would do better to focus on better understanding the texts and developing techniques to deal with their complexities.

For instance, the nature and the meaning of the diacritics that appear in the |Xam texts were not investigated directly. It is assumed that the relatively poor recognition accuracies for the |Xam script and other similarly complex scripts can be attributed to the diacritics. Thus, in future studies it may be worthwhile to further investigate the effect that these diacritics have on recognition and possibly exploit linguistic information to determine the importance of diacritics and their ideal treatment during recognition.

The issue of diacritics is also related to the features used for recognition in this study. Different features are likely to contain different information content and, as such, some are better at describing handwritten text than others. However, some features may be better at describing diacritics that are attached to base characters and that take up a relatively small part of the Cartesian space. Further exploration may allow for additional information about features to be discovered, such as their information content and correlations that might exist between the information content and recognition accuracy. In doing this, it may be possible to determine what information content it is desirable for features to have and, using this information, build hybrid features that maximise this desirable information.

Statistical information about a natural language provides a valuable source of information about the distribution of words and characters in the language and can be used to improve the recognition process. Furthermore, the use of high order language models and statistical information about |Xam and related languages can also assist in furthering our understanding of these languages by means of statistical analysis.

Acknowledgments

This research was partially funded by the National Research Foundation of South Africa (Grant numbers: 85470 and 83998) and University of Cape Town. The authors acknowledge that opinions, findings and conclusions or recommendations expressed in this publication are that of the authors, and that the NRF accepts no liability whatsoever in this regard. The authors would like to thank Pippa Skotnes, Cara van der Westhuizen and Thomas Cartwright for their guidance and assistance in accessing and interpreting the |Xam texts.

7. REFERENCES

[1] M. Abd and G. Paschos. Effective arabic character recognition using support vector machines. *Innovations and Advanced Techniques in Computer and Information Sciences and Engineering*, pages 7–11, 2007.

[2] S. Ali Khayam. The Discrete Cosine Transform (DCT): Theory and Application. *Technical Report. Department of Electrical & Computer Engineering, Michigan State University*, 2003.

[3] J. AlKhateeb, J. Ren, J. Jiang, S. Ipson, and H. El Abed. Word-based handwritten Arabic scripts recognition using DCT features and neural network classifier. *Systems, Signals and Devices*, pages 8–12, 2008.

[4] N. Arica. An Off-Line Character Recognition System For Free Style Handwriting. *MSC Thesis, The Middle East Technical University*, 1998.

[5] J. Chen, H. Cao, R. Prasad, A. Bhardwaj, and P. Natarajan. Gabor features for offline Arabic handwriting recognition. *Proceedings of the 8th IAPR International Workshop on Document Analysis Systems - DAS '10*, pages 53–58, 2010.

[6] N. Cristianini and J. Shawe-Taylor. *An introduction to support vector machines: and other kernel-based learning methods*. Cambridge University Press, 2000.

[7] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. *IEEE Computer Conference on Computer Vision and Pattern Recognition*, 1:886–893, 2005.

[8] A. Fischer, M. Wuthrich, M. Liwicki, V. Frinken, H. Bunke, G. Viehhauser, and M. Stolz. Automatic Transcription of Handwritten Medieval Documents. In *15th International Conference on Virtual Systems and Multimedia*, pages 137–142, Sept. 2009.

[9] S. Günter and H. Bunke. Optimizing the Number of States, Training Iterations and Gaussians in an HMM-based Handwritten Word Recognizer. In *Proceedings of the Seventh International Conference on Document Analysis and Recognition*, volume 1, pages 472–476, 2003.

[10] S. Haboubi, S. Maddouri, N. Ellouze, and H. El-Abed. Invariant Primitives for Handwritten Arabic Script: A Contrastive Study of Four Feature Sets. *International Conference on Document Analysis and Recognition*, pages 691–697, 2009.

[11] N. R. Howe, S. Feng, and R. Manmatha. Finding words in alphabet soup: Inference on freeform character recognition for historical scripts. *Pattern Recognition*, 42(12):3338–3347, Dec. 2009.

[12] C.-W. Hsu, C.-C. Chang, and C.-J. Lin. A practical guide to support vector classification. 2003.

[13] M. Hu. Visual pattern recognition by moment invariants. *IRE Transactions on Information Theory*, 8(2):66–70, 1962.

[14] C. Igel. Improving the Rprop Learning Algorithm. *The 2nd International Symposium on Neural Computation*, pages 115–121, 2000.

[15] U. Marti and H. Bunke. On the influence of vocabulary size and language models in unconstrained handwritten text recognition. *Internation Conference on Document Analysis and Recognition*, pages 260–265, 2001.

[16] D. K. Nguyen and T. D. Bui. On the Problem of Classifying Vietnamese Online Handwritten Characters. In *10th International Conference on Control, Automation, Robotics and Vision*, pages 17–20, 2008.

[17] A. Oliveira, C. Mello, E. Silva Jr, and V. Alves. Optical digit recognition for images of handwritten historical documents. In *Ninth Brazilian Symposium on Neural Networks*, pages 29–29, Oct. 2006.

[18] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[19] L. R. Ragha and M. Sasikumar. Adapting Moments for Handwritten Kannada Kagunita Recognition. In *2nd International Conference on Machine Learning and Computing*, pages 125–129, 2010.

[20] C. Shu. Artificial neural network ensembles and their application in pooled flood frequency analysis. *Water Resources Research*, 40(9):W09301, 2004.

[21] T. Su, T. Zhang, and Z. Qiu. HMM-based system for transcribing Chinese handwriting. In *International Conference on Machine Learning and Cybernetics*, volume 1, pages 3412–3417, 2007.

[22] H. Suleman. Digital libraries without databases: The Bleek and Lloyd collection. In *Proceedings of the International Conference on Asia-Pacific Digital Libraries*, pages 392–403, 2007.

[23] A. H. Toselli, V. Romero, E. Vidal, and L. Rodriguez. Computer assisted transcription of handwritten text images. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 944–948, 2007.

[24] X. Wang, X. Ding, and C. Liu. Gabor filters-based feature extraction for character recognition. *Pattern Recognition*, 38(3):369–379, Mar. 2005.

[25] K. Williams. Creating a handwriting recognition corpus for Bushman languages. *Proceedings of the International Conference on Asia-Pacific Digital Libraries*, pages 222–231, 2011.

[26] K. Williams, S. Manilal, and L. Molwantoa. A visual dictionary for an extinct language. In *Proceedings of the International Conference on Asia-Pacific Digital Libraries*, pages 1–4, 2010.

[27] K. Williams and H. Suleman. Translating handwritten bushman texts. In *10th annual joint conference on Digital libraries*, pages 109–118, 2010.

[28] K. Williams and H. Suleman. Using a hidden Markov model to transcribe handwritten bushman texts. In *Proceeding of the 11th annual joint conference on Digital libraries*, pages 445–446, 2011.

[29] S. Young, G. Evermann, M. Gales, T. Hain, D. Kershaw, X. Liu, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, and P. Woodland. *The HTK Book (for HTK Version 3.4)*. Cambridge University Engineering Department, 2006.