CHAPTER 2

REVIEW OF LITERATURE

In this chapter, the literature review related to the recognition of different scripts has been presented. Both Non-Indic and Indic scripts have been surveyed in detail for the purpose of recognition which has been discussed in section 2.1 and section 2.2, respectively. Further, the survey has been considered on the basis of bi-scripts and multi-scripts as demonstrated in section 2.3 and section 2.4, respectively. Section 2.5 elaborates the existing algorithms such as feature extraction and classification techniques utilized for the present work. In addition, section 2.6 demonstrates the recognition outcomes attained for different scripts, and section 2.7 highlights the research gaps. Finally, the whole chapter has been summarized in section 2.8.

2.1 NON-INDIC SCRIPTS

In this section, literature related to the recognition of various Non-Indic scripts such as Arabic, Chinese, Dutch, Japanese, Latin, Mongolian, Persian, Thai and Uyghur have been presented.

2.1.1 Arabic

Arabic script is utilized to write the Arabic language and it comprises 18 different letters. It is written horizontally from right to left. After the Latin script, it is the world's second most widely used script. There is a lot of work done in Arabic script on character and word recognition. The major issue in the automatic recognition of Arabic words is to segment them into their various constituents. So, to solve this problem, Septi and Bedda (2006) categorized the city names along with a number of related components and then segmented every component into characters. After segmentation, they described the topological features of characters. For classification, they employed Multilayer Perceptron (MLP) classifier. The reported accuracies were between 90.00% and 98.33% on a database of 48 cities of Algeria. For improving the performance of IKRAA (handwritten Arabic word recognition system), Cheikh and Kacem (2007) proposed a recognition system to recognize handwritten Arabic words based on transparent neural network (TNN-DF). For a description of global and local

features, structural features, and Fourier Descriptors (DF) were employed, respectively. To train the letters, parts of words, and words of city names, three NN-MLPs (Neural Network-Multi Layer Perceptrons) were utilized. They evaluated the proposed system on some samples of Tunisian city names (comprising 15 samples for each of the 50 city names) drawn from the IFN/ENIT database, with a recognition accuracy of 77.6%. Kessentini *et al.* (2010) proposed a script-independent multi-stream approach for the recognition of offline handwritten words. This approach combined features, such as density-based features and contour-based features which were fed to Hidden Markov Model (HMM) classifier for recognition purpose. The approach was evaluated on two public datasets, namely, IRONOFF and IFN/ENIT dataset with recognition accuracies of 89.8% and 79.8% respectively, which revealed significant results in comparison to existing word recognition strategies (Tay *et al.*, 2001; Märgner *et al.*, 2005; Viard-Gaudin *et al.*, 2005).

Bouaziz *et al.* (2014) proposed a recognition system for Arabic handwritten words based on segmentation approach in which the words were segmented by detecting ligatures between the characters and analyzing vertical projection profiles. Structural features comprising 128 feature values were fed to the RBF-SVM (Radial Basis Function-Support Vector Machine) classifier. To treat multi-class data, a "one against one" approach was used. They obtained a recognition accuracy of 96.82% based on a database comprising 500 words collected from four writers. Karim and Kadhm (2015a) developed a recognition system for Arabic handwritten words without segmentation. Various features like structural features, statistical features (connected components features and zoning features), and global transformations (Discrete Cosine Transform (DCT) and Histogram of Oriented Gradient (HOG) features) were employed for extracting characteristics from the word images. For the purpose of classification, Neural Network (NN) classifier was used. They evaluated the system on a dataset comprising 2913 handwritten word images with a recognition accuracy of 95.0%. They (2015b) also experimented with the same approach using different kernels of an SVM classifier (like linear, polynomial, and RBF kernels of SVM) and attained 96.31% recognition accuracy using the polynomial kernel of the SVM classifier. Hafiz and Bhat (2016) proposed a hybrid HMM-KNN classification scheme for Arabic Optical Character Recognition which was the first attempt of its type. They used an analytical approach where the first tier employed a Part-of-Arabic-Word

(PAW) based HMM classifier to produce corresponding log-probabilities for each PAW image. The second tier employed k-NN classifier to allocate classes to tier one's emitted PAWs. The experiment was conducted on the IFN/ENIT database and reported 82.67%, 86%, and 94% classification accuracy on three sets comprising 600, 400, and 200 Arabic words, respectively. Jayech *et al.* (2016) presented a segmentation-based approach to recognize offline handwritten Arabic city names using Dynamic Hierarchical Bayesian Network. To extract features, Zernike and Hu's moments were considered which do not vary with respect to rotation, translation, and scaling. To the best of the authors' knowledge, this was the first endeavor to experiment with IFN/ENIT database using Dynamic Bayesian Network. The proposed approach gained 82.0% accuracy using 946 handwritten Tunisian city names with associated postcodes of the IFN/ENIT database.

Khemiri *et al.* (2016) developed a system to recognize offline handwritten Arabic words using three main steps, namely, base estimation, feature extraction, and classification. To estimate baseline, two methods were employed where the first one made comparisons for each word point by point and the second method made comparisons segment by segment based on both manual baseline and IFN/ENIT's ground truth baseline. Then structural features were given as an input to some forms of Bayesian network, namely, Naive Bayes (NB), Tree Augmented Naive Bays Network (TAN), Horizontal and Vertical Hidden Markov Model (VH-HMM), and Dynamic Bayesian Network. Their approach achieved 90.02% accuracy on the IFN/ENIT database (comprising 7881-word samples of 83 classes) by employing VH-HMM. In order to amalgamate the advantages of both the analytical and holistic approaches, Khlif *et al.* (2016) proposed an online handwritten word recognition system in Arabic script based on the combination of both these approaches. In the analytical approach, a feature vector of 254 features comprising online and offline features were extracted from the images, which were then diminished using Sequential Forward Floating Search (SFFS) feature selection approach. After selecting the desirable features, an SVM classifier was employed for the classification task. Whereas in the holistic approach, they extracted directional features, temporal features, and curvature features from the word samples which were then fed to the HMM for the recognition purpose. Then the outcomes from both these approaches were combined based on combination rules. They attained the recognition accuracy of

90.30% on the ADAB dataset, which was reported as superior in comparison to the recognition accuracy attained through analytical approach only (81.33%) and holistic approach only (85.30%). Moubtahij *et al.* (2016) proposed an HMM toolkit-based method to recognize offline handwritten Arabic words by employing statistical features without requiring prior word segmentation. This method was evaluated on the "Arabic-Numbers" dataset (comprising 1905 sentences and 47 words) and attained an accuracy of 80.33%.

Assayony and Mahmoud (2017) extracted statistical Gabor features and Gabor descriptors from the offline handwritten Arabic words based on holistic approach. The considered features were then merged with the Bag-of-features framework to extract the required features. Based on the CENPARMI database, they attained 86.44% recognition accuracy using an SVM classifier with a linear kernel function. Tamen *et al.* (2017) employed various classifiers (like MLP, SVM, and Extreme Learning Machine (ELM)) to recognize Arabic handwritten words. They considered Chebyshev moments (CM) boosted with some statistical and contour-based features (SCF) to extract features from word images; and to detect local features, statistical features were used. Each of the three classifiers was trained with two sets of features (CM and SCF) individually resulting in six classifiers, for which various combination rules were offered. They reported a 96.82% rate of recognition by evaluating the system on the IFN/ENIT database. Gupta *et al.* (2018) proposed a hybrid approach for the recognition of offline handwritten words that integrated the output of SVM classifiers based on three features, namely, Arnold transform-based features, curvature-based features, and Deep Convolutional Neural Network (DCNN) based features. They employed three strategies, namely, vote for the strongest decision, vote for majority decision, and vote for sum of the decisions in order to integrate the output of the SVM classifiers. The proposed approach was experimented on three public datasets, namely, CENPARMI, ISIHWD, and IAM200 with recognition accuracies of 95.23%, 97.16%, and 95.07%, respectively. In order to recognize words handwritten in Arabic/Persian script, Tavoli *et al.* (2018) developed a new feature extraction approach, namely, SGCSL (Statistical Geometric Components of Straight lines). Based on an SVM classifier, they reported recognition accuracies of 67.47%, 80.78%, and 86.22% on three public datasets, namely, Iran-cities (Dehghan *et al.*, 2001), IFN/ENIT (Pechwitz *et al.*, 2002), and IBN SINA (Farrahi *et al.*, 2010) Arabic

datasets, respectively. Ghadhban *et al.* (2020) presented a survey in the field of offline Arabic handwriting word recognition. They demonstrated the different Arabic datasets with certain limitations and also presented the feature extraction and classification techniques utilized in Arabic handwriting recognition. This survey revealed that the maximum results have been reported on the IFN/ENIT database. Hamida *et al.* (2020) utilized the HOG and Gabor filter descriptors in order to extract features from the offline handwritten Arabic words. The experiments were performed on the IFN/ENIT database using a k-NN classifier and attained accuracies of 80.10% and 78.46% based on HOG and Gabor filter descriptors, respectively.

2.1.2 Chinese

Chinese is affiliated to Chinese branch of the Sino-Tibetan language family. Chinese comprises characters that depict both sound and meaning. Chinese words are composed of one or more syllables where each syllable is represented by a single character. It is spoken by approximately 1.3 billion people in the People's Republic of China, the Republic of China, Singapore, and other areas of Southeast Asia [w1]. Dai *et al.* (2007) presented an overview of the features of Chinese characters along with a character recognition system. They provided a summary of important methods and the results of the Chinese character recognition. The corresponding future directions were also provided. In order to recognize Chinese place names, Zhang *et al.* (2009) proposed a two-step process where the first step employed an N-gram model to detect Chinese place name candidates from the corpus. The second step employed a maximum entropy model to recognize and the model was given semantic concept features based on Hierarchical Network of Concept (HNC) for expressing linguistic knowledge. The proposed approach was evaluated on a test set comprising 17,825 Chinese place names in the close-test and 10,065 Chinese place names in the open-test. The system attained 80.46% (precision rate) and 84.19% (F-value) in open-test; and achieved 85.29% (precision rate) and 88.49% (F-value) in close-test. In order to recognize similar handwritten Chinese characters, Lu *et al.* (2017) presented a hybrid model comprising CNN and SVM for feature extraction and classification, respectively. They attained accuracies of 98.55% and 94.23% using HCL2000 and CASIA-HWDB 1.1 datasets, respectively.

2.1.3 Dutch

Dutch utilizes the same Roman alphabet as English. Dutch is a West Germanic language with approximately 24 million native speakers [w2]. It is the third most widely used Germanic language, after English and German. Waard (1995) proposed an approach to optimize substitution cost parameters of minimal edit distance. For the optimization purpose, a discriminative error criterion with a gradient search algorithm was considered. The proposed approach was evaluated in a basic experiment comprising string to string matching problem. They reported an accuracy of 45.0% and 52.0% on simple and complex substitutions, respectively, based on a lexicon comprising 2599 words. Shridhar *et al.* (2002) proposed a lexicon-driven segmentation approach to recognize Dutch city names based on dynamic programming and also analyzed the impact of lexicon completeness on recognition. They applied local chain code histograms of character contour in order to extract significant features. The proposed approach attained an accuracy of 60.0% on a dataset comprising 9726 city names.

2.1.4 Japanese

This language is mainly spoken in Japan with approximately 128 million native speakers (in 2020). Japanese comprises two scripts, namely, Hiragana and Katakana. Hiragana and Katakana include less than 50 letters, which are basically simplified Chinese characters. Chinese characters used in Japanese writing are known as Kanji. There is an existence of over 40,000 Kanji out of which approximately 2000 amount to over 95% of characters that are actually utilized to write text. As spaces do not exist in Japanese sentences, Kanji aid in breaking up the words in order to make them much easier to be read. Maruyama and Nakano (2000) proposed an approach to recognize cursive Japanese words written in Latin characters. For the recognition purpose, they employed an integration of two classifiers, namely, pattern matching based on directional features and HMM. They achieved a first rank recognition accuracy of 56.8% and 59.2% using only pattern matching and only HMM, respectively, based on 5200 templates comprising 52 classes. Then by integration, there was an improvement in first rank recognition accuracy to 68.4% and the cumulative recognition accuracy among the ten best candidates got improved to 92.5%. Liu *et al.* (2002) proposed a lexicon-driven technique to segment and

recognize handwritten Japanese addresses. To attain real-time recognition, a beam search strategy was employed to manage the lexicon matching. They attained a correct recognition accuracy of 83.68% and an error rate of less than 1% by considering experiments on 3589 handwritten letters. Das and Banerjee (2015) presented a geometry-topology based algorithm to recognize Japanese Hiragana characters. The proposed algorithm was invariant to size, rotation, and translation. For evaluation, they considered 6 distinct handwritten specimens of 45 Hiragana characters individually and attained an average accuracy of 94.1% based on minimum Euclidean distance.

2.1.5 Latin/Roman

Latin script comprises 26 letters and is utilized by approximately 70% of the world's population. It is also called Roman script due to its origin in ancient Rome. It is used as the principal method of writing in most Western, Central, and Eastern European languages. Tay *et al.* (2001) proposed a hybrid approach combining Neural Network (NN) and HMM to recognize handwritten words. For describing the features, geometrical features were extracted from images. They evaluated the proposed approach on three databases, namely, IRONOFF, SRTP, and AWS, and reported that the hybrid recognizer gained 96.1% accuracy which was better as compared to the discrete HMM recognizer. Acharyya *et al.* (2013) proposed a handwritten word recognition system based on holistic approach. In order to extract desirable characteristics from the word samples, longest run features were considered which were then fed to MLP classifier for recognition. They evaluated the proposed approach on the CMATERdb1.2.1 dataset by extracting the English words having 12 distinct classes and attained an accuracy rate of 89.90%.

Roy *et al.* (2014) proposed a verification approach for improving handwritten recognition systems. To represent the word images, they employed Marti-Bunke features. For consistent and precise character segmentation, they considered HMM-based forced alignment using the Viterbi algorithm. Then for the purpose of verification, a word hypothesis rescoring scheme corresponding to character segments was done by Deep Belief Network. The experiments were conducted on Rimes (Latin) dataset comprising 59,203 word images. Based on the comparative evaluation, the best accuracy of 91.84%, was obtained by employing the Deep Belief Network-

based verification approach, whereas the MLP-based verification approach gained 91.03% accuracy. Thus, the results revealed that the verification approach based on Deep Belief Network performs better as compared to MLP based system. Patel *et al.* (2015b) demonstrated a segmentation-free approach to recognize offline handwritten English words. The structural features were extracted from the word images which were then fed to Euclidean distance-based k-NN classifier. The experiments were evaluated on a dataset of 300 word specimens that comprises 30 district names of Karnataka state and reported an accuracy of 90%. Dasgupta *et al.* (2016) proposed a holistic approach to recognize offline handwritten words based on directional features which were extracted using Arnold transformation. To the best of the authors' knowledge, this work was the first one to use Arnold transformation to extract the directional features. To classify the word images, an SVM classifier was employed. The proposed approach attained 87.19% accuracy by evaluating the system on the CENPARMI database.

2.1.6 Mongolian

Mongolian is spoken by about 5 million persons in Mongolia, Afghanistan, China, and Russia [w3]. It comprises 26 letters which include 7 vowels, 2 diphthongs, and 17 consonants. It is written vertically from top to bottom and the column order follows left to right order. The segmentation of words in Mongolian script is laborious due to the consolidation of letters of one word to form the vertical support system. Liu *et al.* (2016) developed a recognition approach to recognize online handwritten Mongolian words using a Convolutional Neural Network classifier (MWRCNN) and position maps. In order to extricate desirable features, two feature combination methods, such as MWRCNN with n branches and MWRCNN with one branch, were considered. Then four multiple classifier methods were employed to increase the performance of recognition using a multi-column deep neural network. The methods used were: multi-column MWRCNN based on position maps, multi-column MWRCNN based on different aspect ratio, multi-column MWRCNN based on multiple feature combination, and multi-column MWRCNN based on all the above. The experiments were evaluated on an online handwritten Mongolian word database (MRG-OHMW) comprising 946 classes where each class comprised about 300 samples written by 300 writers and a word recognition accuracy of 93.24% based on hybrid classifiers was

reported which was superior as compared to benchmarking recognition accuracy of 91.20% (Ma *et al.*, 2016).

Wei *et al.* (2019) developed an end-to-end model comprising two Long Short Term Memories (LSTMs) and one attention network in order to recognize offline handwritten Mongolian words. The first LSTM acted as an encoder and the other LSTM acted as a decoder. To provide the connection between encoder and decoder, Deep Neural Network (DNN) based attention network was utilized. The experiments were evaluated on an offline handwritten Mongolian words (MHW) dataset (Fan *et al.*, 2018) with accuracies of 87.68% and 81.12% on two testing sets of MHW, respectively. The results were reported to be better as compared to the state-of-the-art DNN-HMM model (Fan *et al.*, 2018).

2.1.7 Persian/Farsi

Persian or Farsi is a part of the Iranian branch of Indo-European languages having speakers in Iran, Tajikistan, and Afghanistan. This language is spoken by approximately 70 million native speakers and about 110 million people around the world [w4]. Its alphabet comprises 32 letters and the words are written horizontally from right to left. This script is having various features similar to the Arabic script. Hence, a word recognizer of Farsi/Persian can also be employed for recognizing Arabic words. Dehghan *et al.* (2001) proposed a system to recognize handwritten Farsi/Arabic words based on holistic approach. To extract features from the word images, Histogram of chain-code direction features and Kohonen Self Organizing Feature Map (SOFM) were considered. Then the extracted features were given as an input to HMM for recognizing words. They reported an accuracy of 65.0% without rejection on a database comprising more than 17,000 sample images of 198 city names of Iran. Imani *et al.* (2016) developed a holistic approach to recognize offline handwritten Farsi words using HMM as a recognizer engine. For the purpose of recognition, two kinds of gradient features, namely, directional and intensity gradient features were extracted from the words. Then the extracted features were coded using Kohonen self-organizing vector quantization. By evaluating the proposed approach on the FARSA dataset comprising 198-word classes, they achieved a 69.07% recognition accuracy based on directional gradient features.

Arani *et al.* (2019) proposed a segmentation-free approach for the recognition of handwritten Farsi words using the integration of three HMM classifiers trained individually based on three features, namely, black-white transitions, image gradient, and contour chain code features. The integrated outputs of three HMM classifiers were then fed to an MLP classifier in order to recognize. The proposed approach reported recognition accuracy of 89.06% based on the "Iranshahr 3" dataset which was perceived as superior in comparison to individual base classifiers.

2.1.8 Thai

Thai script comprises 44 consonant letters, 15 vowel symbols that merge into at least 28 vowel forms, and 4 tone diacritics. It is written from left to right in a horizontal way. It is spoken mainly in Thailand and also in some other areas like Singapore, the UAE, and the USA. Vichianchai (2011) proposed a segmentation approach for Thai words using Thai-writing structure matching where the writing structure was taken from the words saved in the 1999 Royal Institute Dictionary and levels of Thai-writing. For experiments, the documents considered were associated with newspapers, articles, encyclopedia, Buddhism, laws, the Royal Family's news, non-fictions, interviewing, and general news, and finally, an accuracy of 94.0% was achieved for word segmentation. Mookdarsanit and Mookdarsanit (2020) proposed a Deep Neural Network (DNN) based Thai handwritten script recognition where they employed Convolutional Neural Network (ConvNet) with Deep Belief Network. They evaluated the proposed method on a dataset of 9282 Thai handwritten images that include 87 classes and reported an accuracy of 98.59% using the Gabor filter. The results revealed the better performance of ConvNet based system than state-of-the-art machine learning methods.

2.1.9 Uyghur

Uyghur is a Turkic language which is mainly spoken in the Xinjiang Uyghur Autonomous Region in China. Its alphabet includes 32 characters which are written in cursive style from right to left. Each of the 32 characters has two to four shapes and the position of the character within a word specifies the choice of shape to use. There is no existence of the lower or upper case. Ibrayim and Hamdulla (2015) proposed a segmentation based technique to recognize online handwritten Uyghur words. For the

purpose of recognition, two types of distance measures, namely, adding edit distance and normalized edit distance were considered. For testing purpose, they collected a database comprising 1460 words from different persons and reported an accuracy of 93.17% for the lexicon having size 10 which got reduced to 51.20% for the lexicon of size 1000 using adding edit distance. Using normalizing edit distance, they achieved an accuracy of 94.85% and 62.19% for the lexicons of size 10 and 1000, respectively. In order to provide a public database for online handwriting recognition in Uyghur, Simayi *et al.* (2020) developed a database comprising 1,25,020 specimens of 2030 words gathered from 393 persons. They attained the maximum Character Accurate Rate (CAR) of 94.95% based on the 1D convolutional model and thus provided the baseline reference to carry out further research.

2.2 INDIC SCRIPTS

A lot of papers have been reviewed related to the recognition of various Indic scripts such as Bangla, Devanagari, Gujarati, Gurumukhi, Kannada, Malayalam, Oriya, and Tamil as discussed in the following sub-sections.

2.2.1 Bangla

Bangla script comprises 11 vowels and 39 consonants. It follows left to right order of writing. It is considered the official language of West Bengal state of India and also the national language of Bangladesh. This script is also known as the Bengali script. The letters in the Bangla script are linked to each other by the horizontal line at the top of these letters which is known as "Matra". There are approximately 265 million speakers around the world [w5]. Because of the structure of the Bangla script, there is an issue to segment the overlapped characters, particularly when they are written in cursive style. Pal *et al.* (2009) proposed a lexicon-driven approach to segment and recognize city names handwritten in Bangla script which finds its application in postal automation. The directional features were fed to Modified Quadratic Discriminant Function (MQDF) which was employed to calculate the character likelihood. The proposed system was evaluated on a dataset comprising 8625 handwritten specimens of 84 city names of India in Bangla script and they attained an accuracy rate of 94.08%. Chowdhury *et al.* (2015) developed a segmentation-based approach to recognize online handwritten Bangla words. After segmentation, fuzzy features were

extracted from each segment, and then the characters were recognized and aggregated to form the desired word based on fuzzy linguistic rules. For the evaluation purpose, they considered a database comprising 500 words collected from 10 individuals and reported a recognition accuracy of 77.0%. Adak *et al.* (2016) proposed a holistic approach for the recognition of offline handwritten Bangla words using a hybrid model. The hybrid model comprised Convolutional Neural Network (CNN) as feature vector extractor and Recurrent Neural Network (RNN) as recognizer. They evaluated the proposed system on three databases, namely, public dataset comprising 17,091 words, newly generated dataset comprising 1,07,550 words, and unconstrained dataset comprising 5,223 words with recognition accuracies of 85.42%, 86.96%, and 70.67%, respectively.

In order to select only relevant and discriminant characteristics from the word images, feature selection has a significant role. In this direction, Das *et al.* (2016) presented a feature selection technique i.e. Harmony Search (HS) technique to diminish the features of the approach proposed by Bhowmik *et al.* (2014a) for the recognition of handwritten Bangla words. Bhowmik *et al.* (2014a) extracted a set of 65 elliptical features and attained 85.88% recognition accuracy using an MLP classifier. On the other hand, Das *et al.* (2016) employed the HS feature selection technique to consider only 48 features and reported an accuracy of 90.29% by considering the same database of 1020 handwritten Bangla words. They concluded that the HS-based feature selection technique provides significant results based on the segmentation-free approach in comparison to the existing feature dimensionality reduction approaches such as Particle Swarm Optimization (PSO) and Genetic Algorithm (GA). Sahoo *et al.* (2018) proposed a holistic approach to offline handwritten Bangla word recognition using the negative refraction attribute of light. Based on this attribute, 186 shape-based features were extracted from the word samples, which were then classified in one of the 80 classes of handwritten Bangla city names using five classifiers, namely, Sequential Minimal Optimization (SMO), MLP, Naive Bayes, Simple Logistics and Random Forest. The proposed approach was evaluated on a dataset of 12,000 handwritten Bangla word samples and attained a recognition accuracy of 87.50% based on a hybrid of SMO, Simple Logistics, and CV Parameter Selection embedded with SMO classifier, which outperformed other state-of-the-art features like HOG (Bhowmik *et al.*, 2014b), SCF-116 and SCF-148 (Tamen *et al.*,

2017), Topological (Malakar *et al.*, 2017) and Elliptical features (Bhowmik *et al.*, 2014a). Bhowmik *et al.* (2019) developed an offline handwritten word recognition system in Bangla script using holistic approach. To extract desirable characteristics from the word samples, they considered features based on shapes like Elliptical, Tetragonal, and Vertical pixel density histogram features. Then, SVM and MLP classifiers were employed for the recognition task. For experiments, they proposed a database of 18,000 word samples handwritten in Bangla and named this database CMATERdb2.1.2 which has been made publicly available for the researchers. The best recognition accuracy of 83.64% was achieved based on the SVM classifier which surpassed other state-of-the-art methods evaluated on the same database (Bhowmik *et al.*, 2014a; Dasgupta *et al.*, 2016; Barua *et al.*, 2017; Malakar *et al.*, 2017).

Ghosh *et al.* (2019) developed an offline handwritten Bangla word recognition system using a holistic approach in which they employed gradient-based and modified SCF. To minimize the feature vector, the Memetic Algorithm (MA) based wrapper filter selection technique was proposed. The reduced feature set was given to MLP classifier for the recognition purpose. The experiments were conducted on a dataset of 7500 Bangla words with a recognition accuracy of 93% which was 3.3% superior to the rate based on the original feature vector. Based on the comparison of the MA-based technique with the Genetic Algorithm (GA) feature selection technique, it was indicated that MA attained a better rate of recognition as compared to GA even though GA considered less features comparatively. Sen *et al.* (2020) proposed an online handwritten word recognition system in Bangla script based on an analytical approach that utilized HMM and language model. Based on an analytical approach, they segmented the words into basic strokes, and then based on point, curvature features, and MLP classifier, they recognized the strokes. After that, they constructed HMM using the top 5 stroke recognition choices in order to recognize the words. Then to precise the HMM outputs (if required), the language model was utilized. They evaluated the proposed approach on a dataset of 5500 Bangla words and gained accuracies of 95.4% and 90.3% at stroke level and word level, respectively. They also experimented with the same method to recognize offline handwritten words of Bangla script that were taken from a bank cheque and concluded that the same method is also suitable to recognize offline handwritten words. To reduce the shape and texture-based features, Malakar *et al.* (2020) developed a Genetic Algorithm (GA) based

hierarchical feature selection technique in order to recognize offline handwritten Bangla words. The resulted feature set got reduced by about 28%, which was then fed to MLP classifier for recognition purpose. The experiments were conducted on a dataset comprising 12,000 word specimens with an accuracy of 95.30% which was reported to be 1.28% better than the rate achieved using the original feature vector. The proposed approach also provided better accuracy as compared to other state-of-the-art methods (Bhowmik *et al.*, 2014a; Bhowmik *et al.*, 2014b; Dasgupta *et al.*, 2016; Barua *et al.*, 2017; Malakar *et al.*, 2017).

2.2.2 Devanagari

Devanagari script is utilized to write Sanskrit, Hindi, Nepali, and Marathi languages and it is also known as Nagari script. It comprises 47 primary characters that include 14 vowels and 33 consonants. Its writing direction follows left to right order. This script is employed for other numerous languages due to which it becomes the most widely used writing system in the world. Shaw *et al.* (2008) proposed a holistic approach to recognize handwritten Devanagari words based on HMM. The histogram of chain code direction features was extracted from the word images which were given as an input to HMM to recognize the considered word. For the evaluation purpose, they gathered a database comprising 39,700 word samples of 100 town names of India from 436 distinct writers. They achieved 80.2% (classification rate), 16.3% (misclassification rate) and 3.5% (rejection rate) on the considered dataset. Patil and Ansari (2014) presented a recognition system to recognize online handwritten Devanagari words from where the features were extracted using android technology. Then the extracted features were recognized using HMM. They experimented with the proposed approach on two sets comprising 50 and 100 words collected from distinct writers, with 96% and 94% recognition accuracies, respectively. They concluded that the word comprising up to 3 characters reported a higher recognition accuracy as compared to word up to 5 characters. In order to boost the performance of offline handwritten Devanagari word recognition system, Shaw *et al.* (2015) proposed an amalgamation of information at feature and classifier levels. For the extraction of desirable features, two features, namely, Directional Distance Distribution (DDD) and Gradient Structural Concavity (GSC) features were considered. Two SVMs were trained for DDD and GSC features whose outputs were

then integrated to train another SVM. The experiments were evaluated on a dataset comprising 39,700 samples of handwritten Devanagari words which comprised 100 town names of India and they reported a recognition accuracy of 88.75%. Kumar (2016) presented a segmentation approach to recognize handwritten Devanagari words in which the headline of the word was eliminated to separate the 'consonants' and complete 'matra' or portion of 'matra' over and beneath the headline. To recognize middle portion characters; and upper & lower portion characters, total 281 and 64 features were extracted, respectively, which were then given to MLP classifier for the classification task. They tested the approach on a dataset of 3600 Devanagari words gathered from over 200 writers and attained overall recognition accuracies of 80.8% and 72.0% based on two and six character words, respectively.

Pramanik *et al.* (2018) presented a strategy to detect the number of components of the Devanagari word image. From the detected components, cumulative stretch and shadow based features were extracted. A number of classifiers were employed to determine whether further segmentation of a component is required or not. Among all the considered classifiers, the Random Forest classifier provided the best result with 98.63% accuracy on CMATERdb 1.5.1 dataset. Based on the comparison, this approach provided better results as compared to the approach proposed by Sarkar *et al.* (2011). To recognize Devanagari's ancient handwritten characters, Narang *et al.* (2020) extracted Scale-Invariant Feature Transform (SIFT) and Gabor filter features which were fed into an SVM classifier for classification. To reduce the dimensionality of the features, Principal Component Analysis (PCA) was applied. Based on a dataset of 5484 specimens of Devanagari characters, they attained an accuracy of 91.39% using the 10-fold cross-validation technique.

2.2.3 Gujarati

Gujarati script is customized from the Devanagari script to write the Gujarati language. This language is utilized in states of India like Gujarat, Rajasthan, Madhya Pradesh, Maharashtra, and Karnataka and is also used in few other countries like Fiji, Malawi, Bangladesh, Kenya, Oman, Mauritius, Uganda, South Africa, Singapore, etc. It is spoken by approximately 55.5 million native speakers [w6]. This script includes 35 consonants, 13 vowels and 6 signs, 13 dependent vowel signs, 4 additional vowels for Sanskrit, 1 concurrency sign, and 9 digits. Its writing direction follows left to right

order. Like Hindi or Bangla words, there is no header line in Gujarati words. Patel and Desai (2011) proposed an approach to identify zones of Gujarati words which was the first endeavor of its type. The approach of zone identification was employed to extract modifiers in the upper and lower zones of the word. Then the middle zone was identified due to the detection of the upper zone and lower zone based on the distance transform. This approach was evaluated on a database comprising 250 distinct handwritten Gujarati words with recognition accuracies of 75.2%, 75.2%, and 83.6% for upper, middle, and lower zones, respectively.

Paneri *et al.* (2017) extracted HOG features from the offline handwritten Gujarati words which were then fed into SVM and k-NN classifiers for the purpose of recognition. The experiments were conducted on a database of 2700 specimens corresponding to 10 city names and the maximum recognition accuracy of 85.87% was attained based on the SVM classifier. Naik and Desai (2019) evaluated the online handwritten Gujarati word recognition system based on hybrid features that integrated directional features with curvature data. Using RBF-SVM as a classifier, they attained accuracies of 95.3%, 91.5%, and 83.3% based on 13,000 character samples, 200-word samples, and 90 sentences, respectively.

2.2.4 Gurumukhi

Gurumukhi script is utilized to write the Punjabi language. It comprises thirty-five consonants, six additional consonants that are created by positioning a dot at the foot of the consonant, nine vowel diacritics, three subscript letters, and three auxiliary signs. It follows the left to the right horizontal direction of writing. There are approximately 125 million native speakers of Punjabi language [w7, w8]. In the Gurumukhi script, a lot of work has been done on character recognition. For example, Sharma and Jhajj (2010) proposed an approach to recognize offline handwritten Gurumukhi characters based on zoning features where they gained recognition accuracies of 72.5% and 72.0% based on k-NN and SVM classifiers, respectively. Kumar *et al.* (2013a) proposed a Principal Component Analysis (PCA) based approach by reducing unnecessary features in order to recognize offline handwritten Gurumukhi characters. A number of features like zoning, diagonal, intersection and open-end points, direction, transition, power curve fitting, and parabola curve fitting based features were extracted from the character images which were then given to

classifiers like k-NN, linear SVM, polynomial SVM, and RBF SVM for the purpose of classification of characters. For experiments, a database comprising 16,800 character samples (35 akharas of Gurumukhi) was collected which was partitioned into three groups, each comprising 5600 samples. Based on PCA, they gained recognition accuracies of 99.06%, 98.73%, and 78.30% for each of the considered groups. Kumar *et al.* (2014a) presented a novel hierarchical technique for the recognition of characters handwritten in Gurumukhi script based on three features, namely, centroid features, diagonal features and peak extent features (horizontally and vertically). To minimize the dimensionality of the feature, they considered three feature selection techniques, namely, Consistency-Based (CON), Correlation-based Feature Selection (CFS), and Principal Component Analysis (PCA). Based on PCA, they achieved 91.80% accuracy with linear-SVM classifier by conducting experiments on a dataset of 3500 character specimens.

Kumar *et al.* (2014b) also tested the applicability of two newly proposed feature extraction approaches such as parabola curve fitting and power curve fitting based features to offline handwritten Gurumukhi character recognition system. Based on power curve fitting features, they achieved accuracies of 97.14% and 98.10% based on SVM and k-NN classifiers, respectively, using a dataset of 3500 character specimens. Kumar *et al.* (2016) developed a novel boundary extent feature extraction technique in order to gather specific characteristics from the Gurumukhi handwritten characters which got reduced based on the PCA feature selection approach. For the classification purpose, three classifiers, namely, SVM, k-NN, and MLP were employed. The experiments were evaluated using a 5-fold cross-validation technique on a dataset comprising 7000 samples of characters with a 93.8% recognition accuracy using RBF-SVM classifier. Dhiman and Lehal (2017) proposed an approach to recognize machine printed word images based on DCT and Gabor filter which were used to extract features from the word images. Then these extracted features were fed into the k-NN classifier in order to recognize the considered words. k-NN classifier was trained with 1600 samples of word images and to test the system, 750 word samples were considered. The system reported accuracy rates of 92.62% and 96.99% based on the Gabor filter and DCT, respectively. To recognize offline handwritten Gurumukhi characters, Kumar *et al.* (2018) provided performance comparison of various feature selection techniques, namely, Consistency Based Analysis (CBA),

Correlation Feature Set (CFS), Chi-Squared Attribute (CSA), Independent Component Analysis (ICA), Latent Semantic Analysis (LSA), Principal Component Analysis (PCA) and Random Projection (RP) using two classification techniques, namely, Nearest Neighbour (NN) and linear-SVM. These feature selection techniques were applied to the boundary extent features extracted from the character samples. The experiments were conducted on a dataset of 11,200 character samples and it was concluded that the CSA feature selection technique performed best with maximum recognition accuracies of 88.3%, 95.2%, and 91.3% for upper, middle, and lower zones of Gurumukhi script, respectively, based on NN classifier.

Kumar *et al.* (2019) proposed Adaptive Boosting (AdaBoost) and Bootstrap Aggregating (Bagging) methodologies for the recognition of medieval handwritten Gurumukhi manuscripts based on various feature extraction techniques such as zoning, DCT, and gradient features along with their different combinations. To classify the medieval handwritten characters, four classification techniques, namely, k-NN, SVM, Decision Tree, and Random Forest were used, and they also proposed combinations of these considered classifiers based on the majority voting scheme. By performing experiments on a dataset of 1140 character samples, the maximum recognition accuracy of 95.91% was attained based on AdaBoost methodology and a hybrid of all the considered classifiers by employing the combination of zoning and DCT features. In order to support the research in online handwritten Gurumukhi script words and numerals, Singh *et al.* (2020) developed a benchmark dataset. They released two datasets, namely, OHWR-GNumerals and OHWR-GScript which comprise 10 stroke classes and 95 stroke classes, respectively generated by 190 writers in order to demonstrate the Gurumukhi character set. They made this dataset publicly available for the researchers.

2.2.5 Kannada

Kannada script has been originated from Brahmi script. This script comprises 49 characters which include 34 consonants (vyanjana), 13 vowels (swara) and 2 yogavahakas. It is considered as the official language of Karnataka state of India and is used by approximately 56 million speakers in the world [w9]. Patel and Reddy (2014) proposed a grid based approach to recognize offline handwritten Kannada words where each word was partitioned into four grids. Then subspace learning

approach, i.e., PCA was exercised on each grid for effective representation. Based on the extracted features, the words were classified based on the Euclidean distance measure. The proposed approach gained a recognition accuracy of 68.57% by considering a dataset of 28 district names of Karnataka and concluded that the proposed method based on the grid approach with subspace learning is superior as compared to the standard PCA method. Patel *et al.* (2015a) proposed an approach to recognize offline handwritten Kannada words based on Locality Preserving Projections (LPP) feature extractor. Then the words were classified using an SVM classifier and the gained results were compared with the k-means classifier. The proposed approach was evaluated on a dataset comprising handwritten words of 30 districts and 174 taluk names of Karnataka state gathered from 50 distinct persons. They reported an average recognition accuracy of 85.0% based on the SVM classifier which was superior as compared to the average recognition accuracy of 83.0% achieved by the k-means classifier. Gowda *et al.* (2017) employed an LPP feature extractor to extract features which were given to the SVM classifier for the purpose of recognition. They achieved an average recognition accuracy of 80% on a dataset comprising words of 30 district names of Karnataka written by 20 distinct individuals.

2.2.6 Malayalam

Malayalam script comprises 95 characters that include 36 consonants, 13 vowels, 12 vowel signs, 5 chillu, 4 consonant signs, numbers and the rest are the anuswaram. It is considered the main language of the Kerala state of India. It follows horizontal left to right order of writing. Kumar and Chandran (2015) proposed a segmentation-based approach to recognize handwritten Malayalam words based on direction features. The extracted features were employed in the MLP neural network for the purpose of classification. Then the recognized characters were assembled to model a word. Jino and Balakrishnan (2017) proposed an approach to recognize offline handwritten Malayalam words based on wavelet transforms and RBF-SVM classifier. The wavelet coefficients were diminished using PCA. They attained a recognition accuracy of more than 90% based on a dataset of 736 word specimens. Jino *et al.* (2019) proposed a recognition system for offline handwritten Malayalam words using a holistic approach based on CNN as feature extractor and SVM as classifier. The proposed approach was experimented on a dataset of 10,676 handwritten word specimens that

correspond to 314 word classes and the system attained 96.90% accuracy. The proposed approach was also evaluated on databases of Marathi and Hindi legal amount words (Jayadevan *et al.*, 2011) and provided better results as compared to state-of-the-art methods.

2.2.7 Oriya

Oriya script includes 41 consonants and 11 vowels with writing directions from left to right. There is no notion of upper/lower case in this script. This script originated from the Kalinga script and is considered the official language of Odisha and the second official language of Jharkhand state of India. Chaudhuri *et al.* (2002) proposed a recognition system to recognize printed Oriya script. To recognize individual characters, hybrid of stroke and run-number based features accompanied by water reservoir based features were employed. Then the characters were classified based on tree classifier and template matching approach with a recognition accuracy of 96.3%. Tripathy and Pal (2004) proposed a segmentation approach to segment the handwritten text of Oriya script into lines, words, and characters. The text lines were segmented into words based on vertical projection profile and structural features. Finally, the words were segmented into characters on the basis of topological, structural, and water reservoir concepts. The proposed word segmentation approach reported an accuracy rate of 98.2% by considering a dataset of 3700 words whereas the isolated and connected characters were identified with an accuracy rate of 96.7% on a dataset comprising 3200 components comprising 2150 isolated and 1050 connected characters. Rushiraj *et al.* (2016) proposed a recognition system for handwritten Oriya characters based on geometric features and Euclidean distance measure. An accuracy of 87.6% was attained based on a dataset of 720 characters.

2.2.8 Tamil

Tamil script is utilized to write the Tamil language which is used in Tamil Nadu state of India, Singapore, Sri Lanka, Malaysia, Canada, USA, and many other countries. This script comprises 18 consonants, 12 vowels, and one special character. It follows left to right order of writing in a horizontal way. Thadchanamoorthy *et al.* (2013) proposed a recognition approach to recognize offline city names handwritten in the Tamil language based on directional features that were supplied to the Modified

Quadratic Discriminant Function (MQDF). For experimentation, they also proposed a database comprising 26,500 word samples which include 265 classes of city names. The proposed approach reported a recognition accuracy of 96.89% on the considered dataset.

2.3 BI-SCRIPTS

In certain situations, there is a need to process Bi-lingual documents that employ two languages (one is the national language of the country and another one is the foreign language), for example, on postal documents. A lot of work on the identification of bi-scripts has been reported in the literature. For example, Roy and Pal (2006) proposed an approach to identify Oriya and Roman scripts at the handwritten word level. A number of features like the presence of small components, water reservoir-based features, topological features, and fractal-based features, etc. were extracted from the word images which were then given to Multilayer Perceptron Neural Network (MLP-NN) for classification of considered words. By experimentation on a dataset of 2500 handwritten words comprising 1200 Oriya and 1300 English words, an accuracy of 97.69% was achieved. Dhandra *et al.* (2007) proposed an approach to identify the scripts from two printed bilingual documents of Devanagari and Kannada comprising printed and handwritten English (Roman) numerals at word level based on morphological reconstruction. The segmentation process was applied to the documents in order to extract the word samples. Various shape-related and directional stroke features were extracted which were then fed to the k-NN classifier for classifying the word samples. They tested the proposed approach on a dataset of 2250 word samples comprising Devanagari, Kannada, and English numerals and attained identification rates of 96.10% (printed Kannada words and Roman numerals), 98.61% (printed Devanagari words and Roman numerals), 94.2% (printed Kannada, Devanagari words and Roman numerals), 92.89% (printed Kannada words and handwritten English numerals) and 98.53% (printed Devanagari words and handwritten Roman numerals). To the best of their knowledge, this task was the first of its type.

Roy *et al.* (2010) proposed an approach at word level for identification of handwritten Persian and Roman scripts which was the first endeavor of its type. For feature extraction, they considered 12 features comprising 4 features based on fractal

dimension, 5 features based on topology, and the rest 3 features based on the position of small components concerning word image. On the basis of these considered features, words were classified using multiple classifiers such as MLP neural network, k-NN, SVM, and MQDF by considering a dataset of 5000 handwritten words comprising 2423 Roman and 2577 Persian words. Out of all the considered classifiers, SVM based on Gaussian kernel achieved the best identification rate of 99.20%. Bianne-Bernard *et al.* (2011) presented an approach to recognize handwritten words of Latin and Arabic scripts based on HMM that took into account dynamic and contextual features. Experiments were conducted on three public datasets, namely, IAM, Rimes, and OpenHart. They attained the best recognition accuracy of 89.1% on the Rimes dataset using the Neural Network combination strategy. Rani *et al.* (2013) proposed a zoning approach to identify Roman numerals and words from Gurumukhi script which was the first task of its type. To extract desirable features from the word samples, Gabor filters were considered and then an SVM classifier with different kernel functions was employed for the recognition purpose. Experiments were conducted on a database comprising 11,400 words which include 1900 Roman numerals, 4288 Roman words, and 5212 Gurumukhi words with accuracy rates of 92.87%, 93.28%, and 99.39% based on linear SVM, polynomial kernel SVM, and RBF SVM classifiers, respectively.

Zinjore and Ramteke (2015) developed an approach to identify Devanagari (Marathi) script and Roman words from printed documents. To identify the scripts, header-line pixel count and inter-character gap features were extracted and then classification was done using a heuristic approach. They reported an identification accuracy of 85.95% by considering a dataset of 10 documents comprising 474 words. In order to recognize online handwritten words of two Indic scripts, i.e., Devanagari and Bangla, Ghosh and Roy (2016) demonstrated a comparative analysis of three feature extraction approaches, namely, directional and structural features, zone wise structural and directional features, and zone wise slopes of dominant points. Then the words based on extracted features were recognized using HMM. They evaluated the proposed system on a database comprising 350 distinct words collected from 100 writers (50 Bangla and 50 Hindi speakers) with accuracy rates of 90.23% and 93.82% for Bangla and Devanagari scripts, respectively. They concluded that the dominant point-based local feature extraction approach generates the best results for both the

scripts. Roy *et al.* (2016) identified the handwritten words in Bangla and Devanagari scripts based on the integration of HMM based holistic approach with Pyramid Histogram of Oriented Gradient (PHOG) feature. In this proposed approach, the word image was partitioned into three zones in a horizontal manner, i.e., upper, middle, and lower zones. Then the middle zone was recognized using HMM, and upper and lower zone modifiers were recognized using an SVM classifier. The final word recognition was done based on the integration of zone-wise results. They evaluated the proposed approach on a dataset comprising 17,091 samples of Bangla words and 16,128 samples of Devanagari script and attained accuracy rates of 92.89% and 94.51%, respectively.

Pramanik and Bag (2020) proposed a segmentation-based approach to recognize handwritten Devanagari and Bangla words. Three types of statistical features got extracted from the considered images and their combinations were fed to 5 classifiers, namely, MLP, SVM, Random Forest, k-NN, and KStar classifiers. In this approach, they proposed Convolutional Neural Network-Transfer Learning (CNN-TL) framework comprising seven architectures, namely, Alexnet, VGG-16, VGG-19, Googlenet, Resnet18, Resnet50, and Densenet201, and then made a comparison of this proposed approach with hand-made features (topographical characteristics of the local segments of the whole image; and global characteristics of the whole image). This approach was tested on four datasets, namely, Cmaterdb 1.1.1 (Sarkar *et al.*, 2012), Cmaterdb 1.5.1 (Singh *et al.*, 2017), ICDAR 2013 Segmentation dataset (Stamatopoulos *et al.*, 2013), and PHDIndic_11 dataset (Obaidullah *et al.*, 2018) from where they extracted 2000 Bangla and 2000 Devanagari word samples for the experiments. Using a hybrid of hand-made features, accuracies of 87.52% and 96.67% were attained with MLP and Random Forest classifiers for Bangla and Devanagari words, respectively. Whereas, CNN-TL framework attained 95.14% and 97.37% accuracies for Bangla and Devanagri words, respectively, which surpassed the accuracies attained through hand-made features.

2.4 MULTI-SCRIPTS

Pal *et al.* (2012) proposed a lexicon-driven approach to recognize tri-lingual city names i.e. English, Hindi, and Bangla, which finds its application in postal automation. Dynamic Programming was utilized to integrate the primitive

components into characters. To the best of the authors' knowledge, this was the first work of its type that considered tri-lingual city names to recognize. The proposed approach attained a recognition accuracy of 92.25% by considering evaluation on a database comprising 16,132 Indian city names that comprised 4257, 8625, and 3250 samples of Hindi, Bangla, and English city names, respectively. In order to remove the problem of scripts having less training data to recognize handwriting, Bhunia *et al.* (2019) developed the Adversarial Feature Deformation Module (AFDM) to boost the system performance by learning explanatory features. They utilized Convolutional Recurrent Neural Network (CRNN) (Shi *et al.*, 2017) and PHOCNet (Sudholt and Fink, 2016) for handwritten word recognition and word spotting, respectively. They evaluated the proposed approach on two Indian scripts such as Bangla and Devanagari and two Latin script databases like Rimes and IAM databases and attained WER (Word Error Rate) of 10.47% on the RIMES dataset which was significant by experimenting on less training set. According to them, this task was the first one to use an adversarial learning approach to handwritten word recognition and spotting.

Bhunia *et al.* (2020) developed an offline-online multi-modal deep neural network in order to consider the data from both offline and online modes to identify the scripts. They designed a Convolutional-LSTM framework where the Convolutional network extracted sequential features and the LSTM component extracted contextual information from the data. They evaluated the approach on 7 scripts, namely, Devanagari, Oriya, Gurumukhi, Bangla, Tamil, Telugu, and English, and achieved average accuracy of 98.17% on the basis of words, which was superior in comparison to other state-of-the-art basic methods (Jaeger *et al.*, 2001; Graves *et al.*, 2008; Fernández-Delgado *et al.*, 2014; Chherawala *et al.*, 2016; Roy *et al.*, 2017). They also considered this work as the first of its type to combine both offline and online modes in a single deep network.

2.5 ALGORITHMS USED IN THIS WORK AT DISTINCT PHASES OF RECOGNITION SYSTEM

An offline handwritten word recognition system comprises various phases, namely, digitization, pre-processing, segmentation, feature extraction, feature selection, classification, and post-processing as described in section 1.2. For the present work, we have considered a holistic approach (segmentation-free approach) to word

recognition which is described in chapter 4. The feature selection techniques utilized for this work are discussed in chapter 5. The rest of the phases and their associated algorithms, if any, are discussed in the following sub-sections.

2.5.1 Digitization

In the digitization phase, a digital image of a paper document is generated through a scanning process. The paper document is scanned at 300 dpi (dots per inch) resolution using a scanner that works by beaming light at the document in order to get the digital image of the document. The digitized image of the document is then fed to the pre-processing phase.

2.5.2 Pre-processing

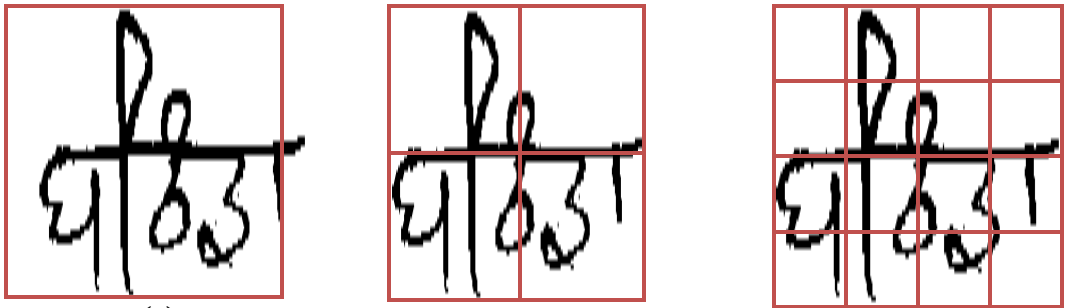
In the pre-processing phase, the word image gets normalized into a window of size 256×64 in order to provide uniformity to all the word images. After normalization operation, the word image is converted into the bitmap image which is then transformed into the thinned image using the parallel thinning algorithm suggested by Zhang and Suen (1984).

2.5.3 Feature Extraction

In order to extract the most significant features from the word image, various features extraction techniques, namely, zoning features, centroid features, diagonal features, intersection & open-end points features, and peak extent features have been utilized, which are explored in the following sub-sections.

2.5.3.1 Zoning features [Rajashekararadhya and Ranjan, 2008]

In a recognition system, the zoning feature extraction technique is one of the most prominent methods to extricate the features from the character or word image. In this feature extraction technique, the word image is divided into a number of zones as illustrated in Figure 2.1. Let L depict the current level of the image. At the current level, the considered number of zones is $4^{(L)}$. Then from each level, the foreground pixels corresponding to $4^{(L)}$ zones are obtained. These foreground pixels get normalized to [0,1] to get the feature set. For the present work, we have considered L=0, 1, 2, and 3.



(a) (b) (c)

Figure 2.1. Digitized word image (a) at level L=0 (b) at level L=1 (c) at level L=2

2.5.3.2 Centroid Features [Basu *et al.*, 2009]

The central position of a two-dimensional word image is considered as the centroid having (*x*,*y*) coordinates. After partitioning the word image into a number of zones, the centroid coordinates (*x*,*y*) of foreground pixels from each zone of the word image are taken as features.

The following procedure has been used to extract the centroid features:

I. Split the thinned image of a word into $4^{(L)}$ number of zones, where each zone is having a size of 8×8 pixels.

II. Obtain the foreground pixel coordinates from each zone.

III. Evaluate the centroid of the foreground pixels and save the centroid coordinates as a feature value.

IV. The feature value of those zones is taken as zero which does not contain any foreground pixel.

The above steps generated a feature set comprising $2\times4^{(L)}$ elements at each level L.

2.5.3.3 Diagonal Features [Pradeep *et al.*, 2011]

In character and word recognition, diagonal features are considered as significant to attain higher accuracy in recognition and to reduce the rate of misclassification. To attain the diagonal features, the word image is partitioned into a number of uniform-sized zones and then foreground pixels are obtained from each diagonal as features of that diagonal. As illustrated in Figure 2.2 (b), the diagonal features are extracted from zone 21 i.e. Z_{21} corresponding to Figure 2.2 (a).

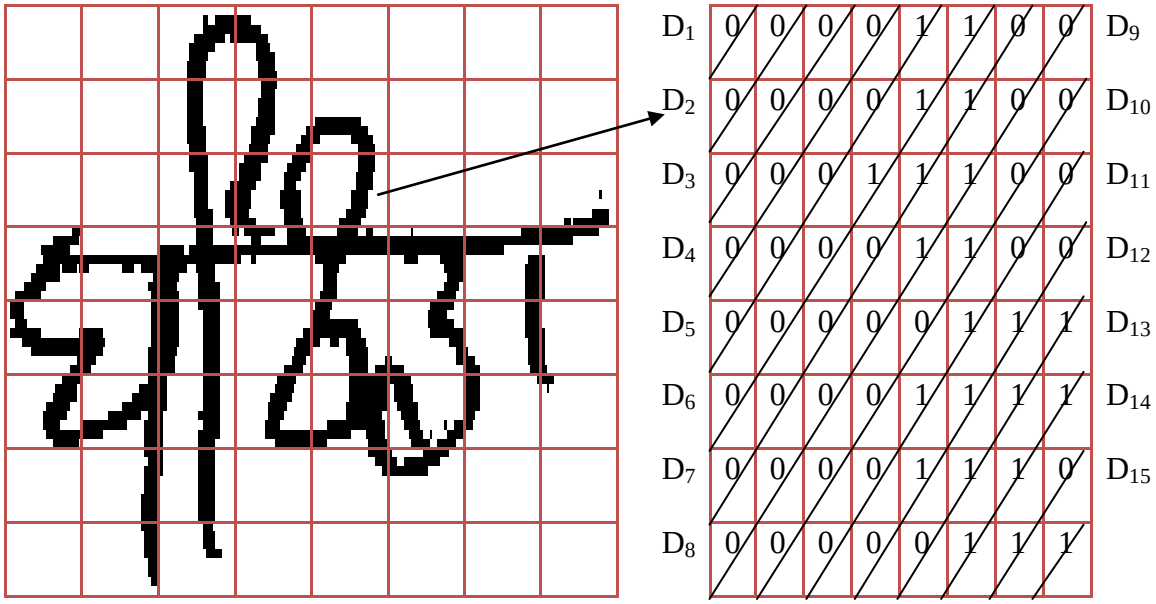


Figure 2.2. (a) Zones of word image (b) Diagonals of Z_{21} zone

The following procedure has been used to extract the diagonal features:

I. Split the thinned image of a word into $4^{(L)}$ number of zones, where each zone is having a size of 8×8 pixels.

II. Each zone comprises 15 diagonals; a single sub-feature value is attained based on the addition of the foreground pixels that exist along each diagonal.

III. The single value is attained by averaging the 15 sub-feature values, which is considered as the feature of that particular zone.

IV. The feature value of those zones is taken as zero which does not contain any foreground pixel along the diagonal.

The above steps generated a feature set comprising $4^{(L)}$ elements at each level L.

2.5.3.4 Intersection & open-end points features [Arora *et al.*, 2008]

An intersection point is considered as the pixel that comprises more than one pixel as its neighbors. An open-end point is considered as the pixel that comprises only one pixel as its neighbor. The intersection & open-end points are illustrated in Figure 2.3 (b) corresponding to zone 4 i.e. Z_4 of considered word image (Figure 2.3 (a)).

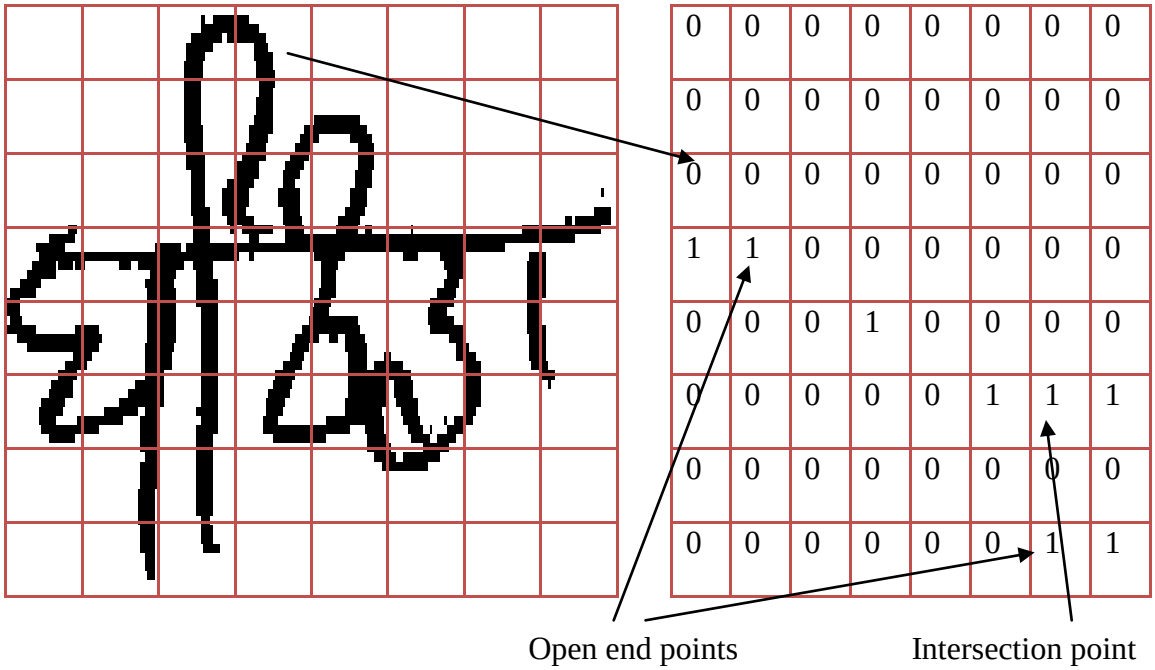


Figure 2.3. (a) Zones of word image (b) Intersection & open-end points

The following procedure has been used to extract the intersection & open-end points features:

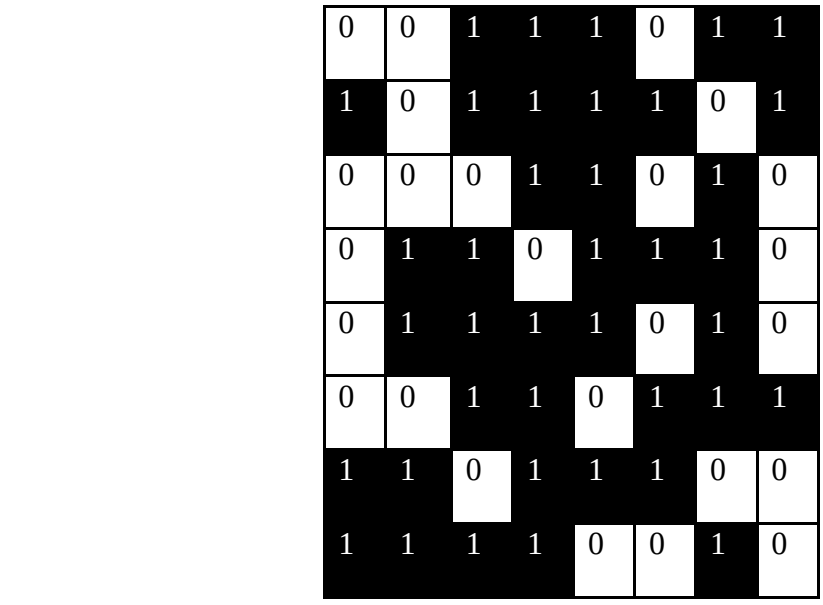
I. Split the thinned image of a word into $4^{(L)}$ number of zones, where each zone is having a size of 8×8 pixels.

II. Corresponding to each zone, compute the number of intersection & open-end points.

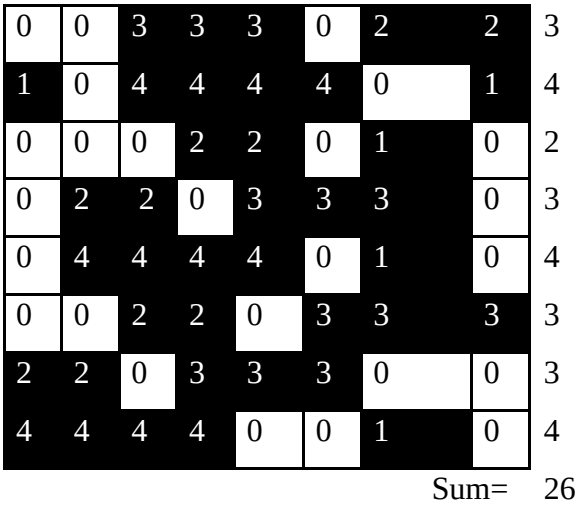
The above steps generated a feature set comprising $2\times4^{(L)}$ elements at each level L.

2.5.3.5 Peak extent features [Kumar *et al.*, 2013b]

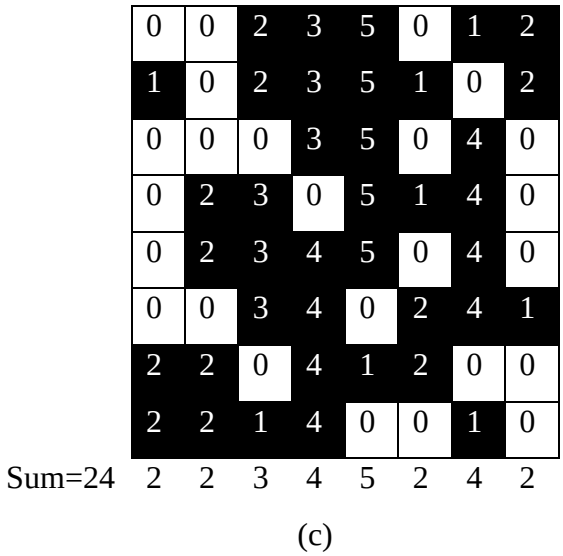
The peak extent features are extricated by considering the addition of the peak extents that place successive black pixels through each zone. To extract these features, both horizontal and vertical ways are considered. In the horizontal peak extent feature extraction approach, the addition of the peak extents is considered horizontally that place the successive black pixels in each row of a zone as demonstrated in Figure 2.4 (b). In the vertical peak extent feature extraction approach, the addition of the peak extents is considered vertically that place the successive black pixels in each column of a zone as demonstrated in Figure 2.4 (c).



(a)



(b)



(c)

Figure 2.4. (a) Zoning of bitmap word image (b) Horizontal peak extent features (c) Vertical peak extent features

The following procedure has been used to extract the horizontal peak extent features:

I. Split the thinned image of a word into $4^{(L)}$ number of zones, where each zone is having size of 8×8 pixels.

II. Evaluate the peak extent horizontally by considering the addition of successive foreground pixels in each row of a zone.

III. Substitute the values of successive foreground pixels with peak extent value in each row of a zone.

IV. Notice the largest peak extent value in each row.

V. Each zone contains 8 horizontal peak extent features (Figure 2.4(b)). Find the sum of these 8 horizontal peak extent sub-feature values and save this addition as a feature of the corresponding zone.

VI. The feature value of those zones is taken as zero which does not contain any foreground pixel.

VII. Compute the normalized values in the feature vector by dividing each feature vector element by the maximum value in the feature vector.

Similarly, we can find the vertical peak extent features by taking the sum of the peak extents in each column of a zone. These steps generated a feature set comprising $2\times4^{(L)}$ elements at each level L.

2.5.4 Classification

The classification phase determines the class of the considered word image on the basis of extracted/selected features. In order to perform classification, various classifiers, namely, k-NN, SVM, Decision Tree, and Random Forest have been employed, which are explored in the following sub-sections.

2.5.4.1 k-NN classifier [Mucherino *et al.*, 2009]

k-NN or k-Nearest Neighbor is a supervised learning algorithm which finds its applications in various fields like data mining, pattern recognition, intrusion detection etc. k-NN functions by storing all the available cases in the form of training samples and takes a decision on the class of the new case i.e. test sample on the basis of the similarity measure. As a similarity measure between the training sample and the test sample, the Euclidean distance function is utilized. k-NN takes a decision regarding the class of the test sample on the basis of a majority vote of its neighbors, calculated by the distance function. So, *k* plays a significant role in k-NN because it determines the number of nearest neighbors to the test sample. Based on two samples, *x* and *y*, the Euclidean distance function is computed as follows:

$$\text{Dist}(x,y) = \sqrt{\sum_{i=1}^{N}(x_i - y_i)^2} \tag{2.1}$$

where, N implies the total number of features of feature set, x_i refers to the feature vector stored in the library and y_i refers to the candidate feature vector.

2.5.4.1.1 Working of k-NN classifier

Consider a graph plotted in Figure 2.5 in which the orange and blue colored circles are plotted, which correspond to class A and class B, respectively. Consider the star as a new case and the task is to predict whether the star belongs to class A or class B. Based on the k-NN algorithm, the first thing to do is to consider the value of *k* that is 3 in this example. Based on Euclidean distance, the three neighbors closest to the star are computed which are 1 blue and 2 orange circles. Thus, the test sample i.e. the star is classified in the class of orange (class A) on the basis of the majority voting scheme.

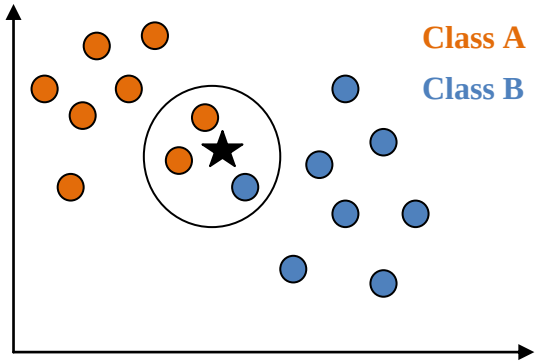


Figure 2.5. Configuration of k-NN classifier

2.5.4.1.2 Features of k-NN classifier

- It is very easy to understand this algorithm and it is simple in its implementation.
- As it doesn't assume any data, it is convenient for non-linear data.
- It can be employed for both classification and regression problems, thus making it a versatile algorithm.
- It requires higher storage as compared to other supervised machine learning techniques.

- As it determines the distance between all the training samples and the testing sample, it is computationally expensive.
- It is vigorous to the noisy natured training data.

2.5.4.2 SVM classifier [Yue *et al.*, 2003]

SVM stands for Support Vector Machine which is considered as one of the supervised machine learning algorithms. SVM aims at separating the *n*-dimensional space (illustrating features) into classes based on some decision boundary or hyperplane. The data points adjacent to the hyperplane are called support vectors because these points provide support to the hyperplane for segregation purposes. Consider Figure 2.6 that segregates the two distinct categories (drawn as circles and rectangles) based on the hyperplane. The two lines are drawn parallel to the hyperplane which is very much near to the considered class. Then the two distances are considered d^- (negative distance) and d^+ (positive distance) which refer to the distance between the hyperplane and the lines parallel to it, respectively. The sum of these two distances is considered as the margin which plays a key role in deciding the hyperplane.

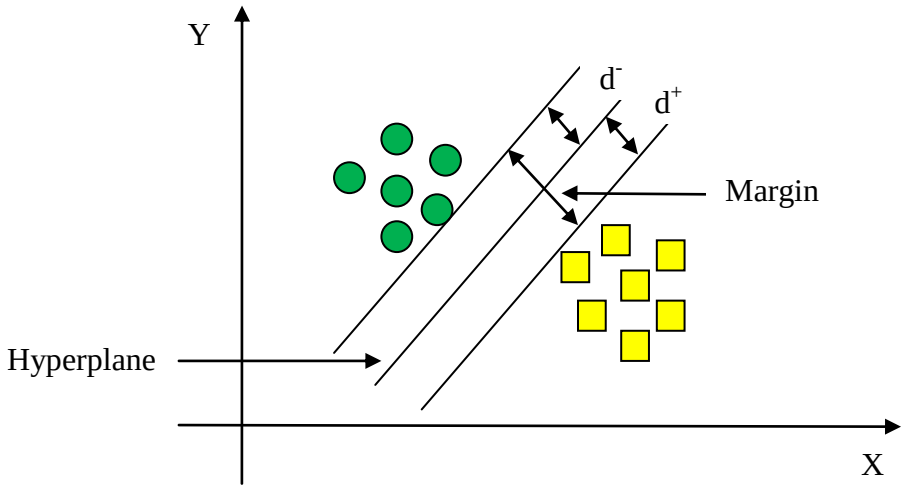


Figure 2.6. Configuration of SVM classifier

2.5.4.2.1 Types of SVM classifier

- *Linear SVM*: Using a linear SVM, data is segregated into two classes by plotting a straight line.
- *Non-linear SVM*: Using a Non-linear SVM, data can't be segregated using a straight line. Instead the data is mapped into high dimensional space for classification. Kernels are employed to convert non-separable data into

separable data. There are multiple kernels available like **Polynomial Kernel, Radial Basis Function (RBF), Gaussian Kernel, Sigmoid Kernel, Laplace RBF Kernel etc.**

2.5.4.2.2 Features of SVM classifier

- It requires less memory due to the utilization of a subset of training points.
- It provides significant accuracy and functions well in high dimensional space.
- It offers stability because a minor change in data doesn't substantially affect the decision boundary or hyperplane.
- It requires a large time for training purposes and thus, is not appropriate for large datasets.
- It doesn't perform well in case of overlapping classes.
- As compared to Decision Tree, it is laborious to understand by persons.

2.5.4.3 Decision Tree classifier [Kotsiantis, 2013]

Decision Tree is one of the supervised machine learning algorithms that functions by building a tree like structure that comprises a succession of test questions and conditions. The test conditions occur at the root and internal nodes of the tree which differentiate the records having distinct features. The class labels are assigned to all the terminal nodes of a tree as depicted in Figure 2.7. Each branch depicts a decision rule. This algorithm initiates from the root node and then on the basis of the outcome of the test condition, follows the suitable branch. It results into either another internal node which applies a new test condition or to a terminal (leaf) node. The hierarchical structure terminates at the leaf node, where the class labels are assigned to the record and thus, the record is classified in one of the classes.

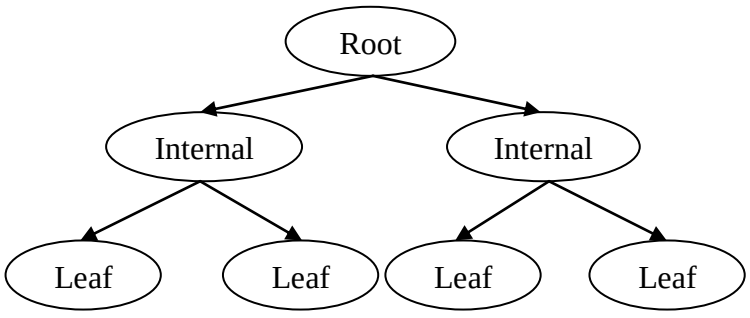


Figure 2.7. Decision Tree

2.5.4.3.1 Features of Decision Tree classifier

- It doesn’t require complex computations to perform classification.
- It is suitable to work with both categorical and continuous variables.
- The procedure to build a Decision Tree is computationally expensive.
- Having multiple classes and a relatively less number of training samples lead to errors in classification.
- It can be utilized to predict missing values and appropriate for variable selection.
- Due to its non-parametric nature, this algorithm doesn't consider the assumptions regarding the distribution.

2.5.4.4 Random Forest classifier [Breiman, 2001]

Random Forest is an ensemble supervised learning technique that is constructed from multiple Decision Trees. It is called a Random Forest because it chooses subsets from the training set randomly in order to construct a group of Decision Trees as depicted in Figure 2.8. As each Decision Tree predicts the class of the test sample, thus the final class of the test sample is predicted on the basis of the majority voting scheme.

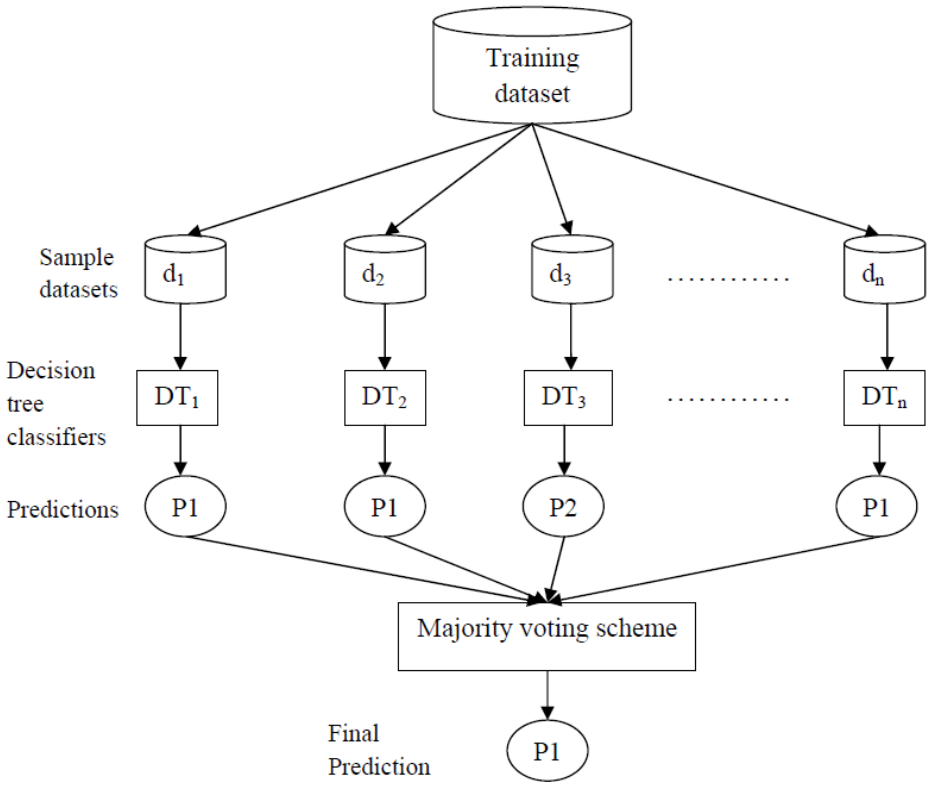


Figure 2.8. Configuration of Random Forest classifier

2.5.4.4.1 Features of Random Forest classifier

- Due to the participation of multiple Decision Trees, this algorithm is observed as more accurate and robust technique.
- It considers the average of all the predictions of the Decision Trees, which abolishes the biases. Thus, it eliminates the over-fitting issue.
- It is suitable to work with missing values by preserving good accuracy.
- It can be utilized for both regression and classification problems.
- It functions well for a large collection of data items as compared to the single Decision Tree.
- As it predicts the final outcome based on the majority voting scheme that considers the predictions of all the considered Decision Trees, it results in the slow process of making a final prediction.

2.6 RECOGNITION ACCURACY ATTAINED FOR INDIC AND NON-INDIC SCRIPTS

In this section, we have demonstrated the reports regarding word recognition of various Indic and Non-Indic scripts which are presented in Table 2.1 and Table 2.2, respectively. As demonstrated in Table 2.1, Bangla and Devanagari scripts provide a lot of work for word recognition. In Bangla and Devanagari scripts, the maximum accuracies of 95.30% and 98.63% have been attained by Malakar *et al.* (2020) and Pramanik *et al.* (2018), respectively. For Oriya, Malayalam, Tamil, Gujarati, and Kannada, one may observe the recognition accuracies of 98.2%, 96.90%, 96.89%, 91.5%, and 85.0%, respectively. For Gurumukhi printed words, a recognition accuracy of 96.99% has been observed.

Table 2.1. Recognition results for Indic Scripts

Authors	**Script**	**Dataset (Word Samples)**	**Feature Extraction/ Selection Technique**	**Classification Technique**	**Accuracy**
Chaudhuri *et al.* (2002)	Oriya	Printed Oriya documents	Hybrid of stroke and run-number based features; water reservoir based features	Tree classifier and template matching approach	96.3%
Tripathy and Pal (2004)	Oriya	3700	Horizontal histograms; vertical projection profile; topological, structural and water reservoir concept	Segmentation approach	98.2%
Shaw *et al.* (2008)	Devanagari	39,700	Histogram of chain code direction features	HMM	80.2%
Pal *et al.* (2009)	Bangla	8625	Directional features	MQDF	94.08%
Patel and Desai (2011)	Gujarati	250	Zone identification: (i) upper zone (ii) middle zone (iii) lower zone	Distance transform	(i) 75.2% (ii) 75.2% (iii) 83.6%
Thadchana-moorthy *et al.* (2013)	Tamil	26,500	Directional features	MQDF	96.89%

Authors	**Script**	**Dataset (Word Samples)**	**Feature Extraction/ Selection Technique**	**Classification Technique**	**Accuracy**
Bhowmik *et al.* (2014a)	Bangla	1020	Elliptical features	MLP	85.88%
Patel and Reddy (2014)	Kannada	1120	PCA	Euclidean distance	68.57%
Patil and Ansari (2014)	Devanagari	(i) 50 (ii) 100	Android technology	HMM	(i) 96% (ii) 94%
Chowdhury *et al.* (2015)	Bangla	500	Fuzzy features	Fuzzy linguistic rules	77.0%
Patel *et al.* (2015a)	Kannada	Handwritten words of 30 district and 174 taluk names of Karnataka	LPP	SVM	85.0%
Shaw *et al.* (2015)	Devanagari	39,700	DDD and GSC features	SVM	88.75%
Adak *et al.* (2016)	Bangla	(i) 17,091 (ii) 1,07,550 (iii) 5,223	CNN	RNN	(i) 85.42% (ii) 86.96% (iii) 70.67%
Das *et al.* (2016)	Bangla	1020	Harmony Search (HS) technique	MLP	90.29%
Kumar (2016)	Devanagari	3600	Middle portion characters: 281 features; upper & lower portion characters: 64 features	MLP	(i) 80.8% (two character words) (ii) 72.0% (six character words)

Authors	**Script**	**Dataset (Word Samples)**	**Feature Extraction/ Selection Technique**	**Classification Technique**	**Accuracy**
Dhiman and Lehal (2017)	Gurumukhi	2350	(i) Gabor filter (ii) DCT	k-NN	(i) 92.62% (ii) 96.99%
Gowda *et al.* (2017)	Kannada	Words of 30 districts of Karnataka	LPP	SVM	80%
Jino and Balakrishnan (2017)	Malayalam	736	Wavelet transforms; PCA	RBF-SVM	More than 90%
Paneri *et al.* (2017)	Gujarati	2700	HOG features	SVM and k-NN	85.87% (SVM)
Pramanik *et al.* (2018)	Devanagari	CMATERdb 1.5.1	Cumulative stretch and shadow based features	Random Forest	98.63%
Sahoo *et al.* (2018)	Bangla	12,000	Shape-based features	SMO, MLP, Naive Bayes, Simple Logistics and Random Forest	87.50%
Bhowmik *et al.* (2019)	Bangla	CMATERdb 2.1.2	Elliptical, Tetragonal and Vertical pixel density histogram features	SVM and MLP	83.64% (SVM)
Ghosh *et al.* (2019)	Bangla	7500	Gradient-based features and modified SCF; MA based wrapper filter selection approach	MLP	93%

Authors	**Script**	**Dataset (Word Samples)**	**Feature Extraction/ Selection Technique**	**Classification Technique**	**Accuracy**
Jino *et al.* (2019)	Malayalam	10,676	CNN	SVM	96.90%
Naik and Desai (2019)	Gujarati	200 (online)	Directional features with curvature data	RBF-SVM	91.5%
Sen *et al.* (2020)	Bangla	5500	Point, curvature based features	MLP and HMM	95.4% (stroke level); 90.3% (word level)
Malakar *et al.* (2020)	Bangla	12,000	Shape and texture based features; GA based hierarchical feature selection approach	MLP	95.30%

From Table 2.2, it is clear that in Non-Indic scripts, a lot of work has been reported in Arabic script for word recognition. In the Arabic script, the maximum recognition accuracy of 97.16% has been reported by Gupta *et al.* (2018), whereas Septi and Bedda (2006) have attained accuracies between 90.00% - 98.33%. One may observe that the accuracies of 98.59%, 98.55%, 96.1%, 94.95%, 93.24%, 92.5%, 91.84%, 89.06%, and 60%, have been attained for Thai, Chinese, Roman, Uyghur, Mongolian, Japanese, Latin, Farsi, and Dutch scripts, respectively.

Table 2.2. Recognition results for Non-Indic Scripts

Authors	**Script**	**Dataset (Word Samples)**	**Feature Extraction/ Selection Technique**	**Classification Technique**	**Accuracy**
Maruyama and Nakano (2000)	Japanese	5200	Directional features	Pattern matching and HMM	92.5%
Dehghan *et al.* (2001)	Farsi/ Arabic	Over 17,000	Histogram of chain-code direction features and Kohonen SOFM	HMM	65.0%
Tay *et al.* (2001)	Roman	IRONOFF, SRTP and AWS	Geometrical features	NN + HMM	96.1%
Shridhar *et al.* (2002)	Dutch	9,726	Local chain code histograms of character contour	Dynamic programming	60.0%
Septi and Bedda (2006)	Arabic	48 cities of Algeria	Topological features	MLP	90.00% - 98.33%
Cheikh and Kacem (2007)	Arabic	IFN/ENIT	Structural features and Fourier Descriptors	NN-MLPs	77.6%
Zhang *et al.* (2009)	Chinese	Close test: 17,825 Open test: 10,065	Semantic concept features	N-gram model and maximum entropy model	84%
Vichianchai (2011)	Thai	1,00,000	Word segmentation	Thai-writing structure matching	94.0%

Authors	**Script**	**Dataset (Word Samples)**	**Feature Extraction/ Selection Technique**	**Classification Technique**	**Accuracy**
Acharyya *et al.* (2013)	Roman	CMATERdb 1.2.1	Longest run features	MLP	89.90%
Bouaziz *et al.* (2014)	Arabic	500	Structural features	RBF-SVM	96.82%
Roy *et al.* (2014)	Latin	59,203	Marti-Bunke features	Deep Belief Network	91.84%
Ibrayim and Hamdulla (2015)	Uyghur	1460	Shape descriptive features	(i) Adding edit distance (ii) Normalized edit distance	(i) 93.17% (ii) 94.85%
Karim and Kadhm (2015a and 2015b)	Arabic	2913	Structural features, statistical features and global transformations	(i) NN (ii) SVM	(i) 95.0% (ii) 96.31%
Patel *et al.* (2015b)	Roman	300	Structural features	k-NN	90%
Dasgupta *et al.* (2016)	Roman	CENPARMI	Directional features	SVM	87.19%
Hafiz and Bhat (2016)	Arabic	IFN/ENIT	695 features	HMM+ k-NN	94%
Imani *et al.* (2016)	Farsi	FARSA	Directional and intensity gradient features	HMM	69.07%
Jayech *et al.* (2016)	Arabic	IFN/ENIT	Zernike and Hu moments	DHBN	82.0%
Khemiri *et al.* (2016)	Arabic	IFN/ENIT	Structural features	Naive Bayes, Tree	90.02% (VH-HMM)

Authors	**Script**	**Dataset (Word Samples)**	**Feature Extraction/ Selection Technique**	**Classification Technique**	**Accuracy**
				Augmented Naive Bays Network, VH-HMM, Dynamic Bayesian Network	
Liu *et al.* (2016)	Mongolian	MRG-OHMW comprising 946 classes	MWRCNN with n branches and MWRCNN with one branch	MWRCNN based on: position maps, different aspect ratio, multiple feature combination and all the above	93.24%
Moubtahij *et al.* (2016)	Arabic	" Arabic-Numbers" dataset	Statistical features	HMM toolkit	80.33%
Assayony and Mahmoud (2017)	Arabic	CENPARMI	Statistical Gabor features and Gabor descriptors merged with Bag-of-features framework	SVM	86.44%
Tamen *et al.* (2017)	Arabic	IFN/ENIT	CM boosted with SCF features	MLP, SVM and ELM	96.82%
Gupta *et al.* (2018)	Arabic	(i) CENPARMI (ii) ISIHWD (iii) IAM200	Arnold transform, curvature and DCNN based features	SVM	(i) 95.23% (ii) 97.16% (iii) 95.07%

Authors	**Script**	**Dataset (Word Samples)**	**Feature Extraction/ Selection Technique**	**Classification Technique**	**Accuracy**
Tavoli *et al.* (2018)	Arabic/ Persian	(i) Iran-cities (ii) IFN/ENIT (iii) IBN SINA	SGCSL	SVM	(i) 67.47% (ii) 80.78% (iii) 86.22%
Arani *et al.* (2019)	Farsi	Iranshahr 3	Black-white transitions, image gradient and contour chain code features	HMM and MLP	89.06%
Wei *et al.* (2019)	Mangolian	MHW (two testing sets)	LSTM	DNN	(i) 87.68% (ii) 81.12%
Hamida *et al.* (2020)	Arabic	IFN/ENIT	(i) HOG (ii) Gabor filter	k-NN	(i) 80.10% (ii) 78.46%
Mookdarsanit and Mookdarsanit (2020)	Thai	9282	Gabor filter	ConvNet with Deep Belief Network	98.59%

Word recognition also plays a significant role in Bilingual documents (For example, Postal documents) in order to identify the script. In this direction, Rani *et al.* (2013) attained a script identification rate of 99.39% on a dataset comprising Gurumukhi words, English words, and English numerals as demonstrated in Table 2.3. Roy *et al.* (2010) achieved a script identification rate of 99.20% based on Persian and English words.

Table 2.3. Recognition results for Bi-Scripts and Multi-Scripts

Authors	**Script**	**Dataset (Word Samples)**	**Feature Extraction/ Selection Technique**	**Classification Technique**	**Accuracy**
Roy and Pal (2006)	Oriya and Roman	2500	Water reservoir, topological and fractal based features	MLP-NN	97.69%
Dhandra *et al.* (2007)	Devanagari, Kannada, Roman	2250	Shape related and directional stroke features	k-NN	98.61% (printed Devanagari words and Roman numerals)
Roy *et al.* (2010)	Persian and Roman	5000	Fractal dimension, topology, position of small components	MLP, k-NN, SVM and MQDF	99.20% (SVM)
Bianne-Bernard *et al.* (2011)	Latin and Arabic	IAM, Rimes and OpenHart	Dynamic and contextual features	HMM	89.1% (Rimes dataset)
Pal *et al.* (2012)	Roman, Devanagari and Bangla	16,132	Histogram of direction chain code features	Dynamic Programming	92.25%
Rani *et al.* (2013)	Gurumukhi and Roman	11,400	Gabor filters	(i) Linear SVM, (ii)Polynomial kernel SVM (iii) RBF SVM	(i) 92.87%, (ii) 93.28%, (iii) 99.39%
Zinjore and Ramteke (2015)	Devanagari and Roman	474	Header-line pixel count and inter character gap features	Heuristic approach	85.95%

Authors	Script	Dataset (Word Samples)	Feature Extraction/ Selection Technique	Classification Technique	Accuracy
Ghosh and Roy (2016)	(i) Bangla (ii)Devanagari	350 distinct words collected from 100 writers	Directional and structural features, zone wise structural and directional features and zone wise slopes of dominant points	HMM	(i) 90.23% (ii) 93.82%
Roy *et al.* (2016)	(i) Bangla (ii)Devanagari	(i) 17,091 (ii) 16,128	PHOG	HMM and SVM	(i) 92.89% (ii) 94.51%

2.7 RESEARCH GAPS

Based on the study of state-of-the-art work in recognition of various Indic and Non-Indic scripts, the following research gaps were analysed.

- Owing to the diverse writing styles of persons, offline handwritten word recognition is undoubtedly a difficult task.
- Due to the existence of multiple characters in each word of any language, the researchers find word recognition an open problem.
- A number of studies have been undertaken in recognition of Gurumukhi characters but till now, no recognized work is available for word recognition in Gurumukhi script.
- In word recognition, a lot of work has been carried out by taking city names as words. Thus, word recognition finds its application in postal automation to sort the mails automatically according to village/city names. Till now, no postal automation system exists for Gurumukhi script.
- Standardized database plays a significant role in recognition. But due to the absence or no utilization of standardized databases in some scripts, it becomes

difficult to compare the proposed approach with the existing approaches.

- In Non-Indic scripts, Arabic script provides a lot of work for word recognition based on created datasets or public benchmark datasets (such as IFN/ENIT). In Indic scripts, a lot of work in word recognition has been reported for Bangla script with the inclusion of feature selection techniques to improve the system performance (Das *et al.*, 2016; Ghosh *et al.*, 2019; Malakar *et al.*, 2020), but no such work exists in Gurumukhi word recognition.
- The segmentation based approach to word recognition possesses certain issues due to the existence of overlapping and touching characters in a word.

2.8 CHAPTER SUMMARY

This chapter has discussed the literature review related to the recognition of various Non-Indic and Indic scripts. A number of Non-Indic scripts like Arabic, Chinese, Dutch, Japanese, Roman, Mongolian, Persian, Thai, and Uyghur have been explored. We have also examined various Indic scripts like Bangla, Devanagari, Gujarati, Gurumukhi, Kannada, Malayalam, Oriya, and Tamil for the word recognition task. Moreover, the different existing algorithms employed at different phases of word recognition like digitization, pre-processing, feature extraction, and classification have been discussed in detail. The summary of the recognition results attained for various Indic and Non-Indic scripts has been demonstrated. At last, the research gaps have been analyzed based on the literature review.