2

STUDY OF OPTICAL CHARACTER RECOGNITION ALGORITHMS AND TOOLS

- **Introduction**
- **Optical Character Recognition Algorithms**
- **Optical Character Recognition Tools**
- **Creation of Master Dataset for Characters and Digits**
- **Result Analysis of tools for Characters and Digits**
- **Summary of Characters and Digits for Evaluated Tools**
- **Recognition Rate Analysis**
- **References**

CHAPTER 2
STUDY OF OPTICAL CHARACTER RECOGNITION ALGORITHMS AND TOOLS

2.1 Introduction

The handwritten character recognition system is classified as online system and offline system.

Online Recogntion: Online handwriting recognition involves the automatic conversion of text as it is written on a special digitizer or PDA, where a sensor picks up the pen-tip movements as well as pen-up/pen-down switching. This kind of data is known as digital ink and can be regarded as a digital representation of handwriting. The obtained signal is converted into letter codes which are usable within computer and text-processing applications [1].

Offline Recognition: Offline recognition operates on pictures generated by an optical scanner. The data is two-dimensional and space-ordered which means that overlapping characters cannot be separated easily. Offline handwriting recognition involves the automatic conversion of text in an image into letter codes which are usable within computer and text-processing applications. The data obtained by this form is regarded as a static representation of handwriting. Offline handwriting recognition is comparatively difficult, as different people have different handwriting styles [2].

Researcher has studied OCR Algorithms which are used for offline character recogntion.

2.2 Optical Character Recognition Algorithms

Researcher has studied following optical character recognition algorithms.

2.2.1 Template Matching Algorithm

2.2.2 Statistical Algorithm

2.2.3 Structural Algorithm

2.2.4 Neural Network Algorithm

2.2.5 Support Vector Machine

2.2.6 Decision Tree Classifier

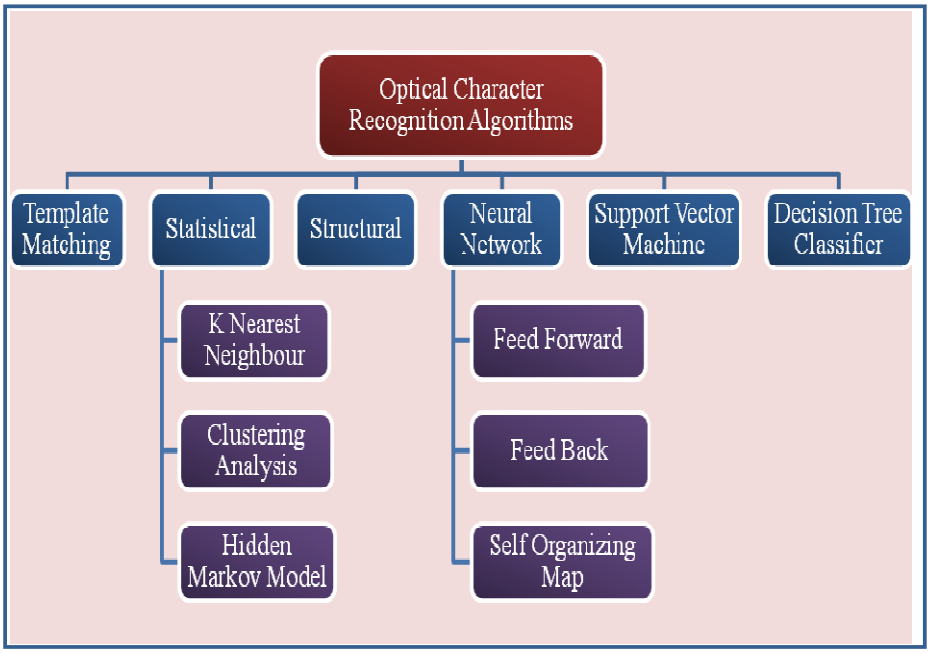

Figure 2.1 Optical Character Recognition Algorithms

2.2.1 Template Matching Algorithm

Template matching is a system prototype that useful to recognize the character or alphabet by comparing two images. Template matching is the process of finding the location of sub image called a template inside an image. Once a number of corresponding templates is found their centers are used as corresponding points to determine the registration parameters. Template matching involves determining similarities between a given template and windows of the same size in an image and identifying the window that produces the highest similarity measure. It works by comparing derived image features of the image and template for each possible displacement of the

template [3]. In Template matching, the character itself is used as a “feature vector”.

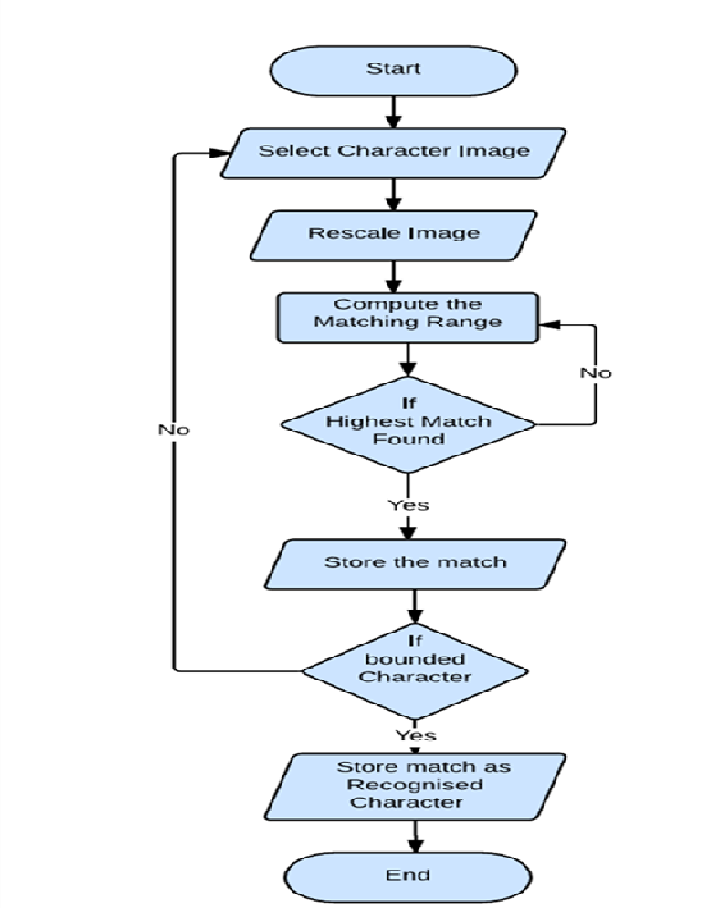

Figure 2.2 Workflow of Template Matching Algorithm

Template matching involves the use of a database of characters or templates. There exists a template for all possible input characters. For recognition to occur, the current input character is compared to each template to find either an exact match or the template with the closest representation of the input character. If *I(x,y)* is the input character, *Tn(x,y)* is the template *n*, then the matching function *S(I,Tn)* will return a value indicating how well template *n* matches the input character. Character recognition is achieved by identifying which *Tn* gives the best value of matching functions, *S(I,Tn).* The method can

only be successful if the input character can be stored templates are of the same or similar font [4].

The template matching algorithm implements the following steps:

I. Firstly, the character image from the detected string is selected.

II. After that, the image to the size of the first template is rescaled.

III. After rescale the image to the size of the first template(original) image, the matching metric is computed.

IV. Then the highest match found is stored. If the image is not match repeat again the third step.

V. The index of the best match is stored as the recognized character.

The value of the data was entered will be extracted from the images, comprising letters. Each character is automatically selected and threshold.

Extraction of the image of the character. The image is converted into 12*12 bitmap.

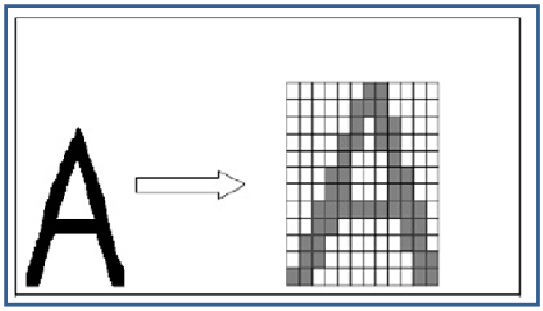

Figure 2.3 Bitmap Image of Character A

Bitmap is representd by 12*12 matrixes or by 144 vectors with 0 and 1 coordinates.

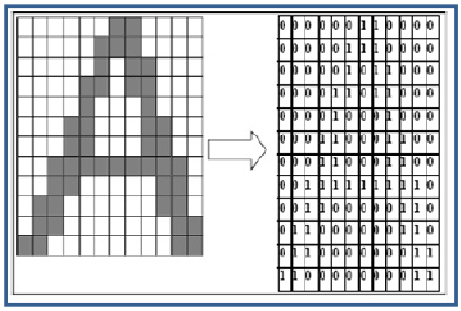

Figure 2.4 12×12 Matrix Representation of Character A

Template matching algorithm is also known as matrix matching or pattern matching algorithm.Template matching algorithm works best with typewritten text but does not work well when new fonts are encountered. It only works on fonts of which it has templates.

❖ **Related work in the Template Matching Algorithm**

Faisal Mohammad et.al. [5] have presented pattern matching algorithm for typewritten and handwritten characters. The binary image is divided into 5 tracks and 8 sectors. The track-sector matrix is then matched with existing template. The existing template consists of each track-sector intersection value, each track value and each sector value. If all these parameters are found to match with the template values then the resultant is the character identified.

Mo Wenying and Ding Zuchun [6] have proposed an improved template matching algorithm which based on the weighted matching degree. After the completion of the pre-processing of input characters, the algorithm uses the moving match of the standard character template with respect to image character template. It uses a method of weighted matching degree. This algorithm avoids the influence of adherent noise and partial distortion, which greatly impacts the recognition rate of the character.

Mr. Danish Nadeem and Miss. Saleha Rizvi [7] have proposed typewritten/handwritten character recognition using template matching. The

aim is to produce a system that classifies a given input as belonging to a certain class rather than to identify them uniquely, as every input pattern. The system performs character recognition by quantification of the character into a mathematical vector entity using the geometrical properties of the character image. Recognition rate of typewritten Standard English alphabets fonts is 94.30%, typewritten Unknown English alphabets fonts is 88.02% and handwritten English alphabets is 75.42%.

Rachit Virendra Adhvaryu [8] has presented template matching algorithm for alphabets. The system prototype has its own scopes which are using Template Matching as the algorithm that applied to recognize the characters, characters to be tested are alphabet (A – Z), and grey-scale images were used with Times New Roman font type and recognizing the alphabet by comparing between two images.

M. Ziaratban et.al. [9] have proposed an approach for character recognition termed as template matching. This technique extracts feature by searching the special templates in input images. For each template, the amount of matching is used as feature and position of the best matching in an image is found and saved [10].

2.2.2 Statistical Algorithm

The purpose of the statistical algorithms is to determine to which category the given pattern belongs. By making observations and measurement processes, a set of numbers is prepared, which is used to prepare a measurement vector [11]. Statistical algorithm uses the statistical decision functions and a set of optimality criteria which maximizes the probability of the observed pattern given the model of a certain class.

Statistical algorihtms are mostly based on three major assumptions:

I. Distribution of the feature set.

II. There are sufficient statistics available for each class.

III. Collection of images to extract a set of features which represents each distinct class of patterns.

The measurements taken from n features of each word unit can be thought to represent an n-dimensional vector space.

There are two approaches of statistical algorithm.

I. Non-parametric Recognition

In Non-parametric Recognition, a priori data or information is not available.

II. Parametric Recognition

Since a priori data or information is available about the characters in the training data, it is possible to obtain a parametric model for each character.

2.2.2.1 Statistical Methods

The major statistical methods applied in the character recognition field are K Nearest Neighbor, clustering Analysis, Hidden Markov Modeling etc.

2.2.2.1.1 K-Nearest Neighbour Algorithm

The k-Nearest Neighbors algorithm (k-NN) is a non-parametric method used for classification. The input consists of the k closest training examples in the feature space. In k-NN classification, the output is a class membership. An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors (k is a positive integer, typically small). If k = 1, then the object is simply assigned to class of that single nearest neighbor [12]. The idea behind k-Nearest Neighbor algorithm is quite straightforward. To classify a new character, the system finds the k nearest neighbors among the training datasets, and uses the categories of the k nearest neighbors to weight the category candidates [13].

The k-NN algorithm can be described using the following equation:

$$y(d_{i)} = \arg\max_{k} \sum_{Xj \in kNN} \mathrm{Sim}\,(d_{i,} x_{j})\; y(\, x_{j,} c_{k}\,)$$

Where, d_i is a test character, x_j is one of the neighbors in the training set, $y(x_j,c_k) \in \{0, 1\}$ indicates whether x_j belongs to class c_k, $Sim(d_i,x_j)$ is the similarity function for d_i. Above equation means the class with maximal sum of similarity will be the winner [13].

The performance of this algorithm greatly depends on two factors, that is, a suitable similarity function and an appropriate value for the parameter k. The similarity function is the Euclidean distance. It is given by below equation.

$$f(x, p^2) = \sum_{i=1}^{N} (x_i - p_i^2)$$

One of the basic requirements for this method to obtain good performance is the access to a very large database or labeled prototype but searching through the whole database to find the nearest objects to a test image is time consuming, and has to be done for every character in a document.

2.2.2.1.2 Clustering Analysis

The goal of a clustering analysis is to divide a given set of data or objects into a cluster, which represents subsets or a group. The partition should have two properties. Homogeneity inside clusters: the data, which belongs to one cluster, should be as similar as possible. Heterogeneity between the clusters: the data, which belongs to different clusters, should be as different as possible [14]. Thus, the characters with similar features are in one cluster. Thus, in recognition process, the cluster is identified first and then the actual character.

2.2.2.1.3 Hidden Markov Modeling

A hidden markov model(HMM) is a statistical model in which the system being modeled is assumed to be a Markov process with unobserved state. The Hidden Markov Model is a finite set of states, each of which is associatd with a probability distribution. Transitions among the states are governed by a set of probabilities called transition probabilities. In a particular state an outcome or observation can be generated, according to the associated probability distribution. It is only the outcome, not the state visible to an external observer and therefore states are "hidden" to the outside; hence the name Hidden Markov Model [15].

A HMM can be represented by a Finite State Machine, which in turn can be represented by either a connected graph or a special form of connected graph called a trellis. Each node in this graph represents a state, where the signal being modeled has a distinct set of properties and each edge a possible transition between two states at consecutive discrete time intervals [16]. An example of a trellis and graph of a 4 state fully connected HMM is shown in Figure 2.5.

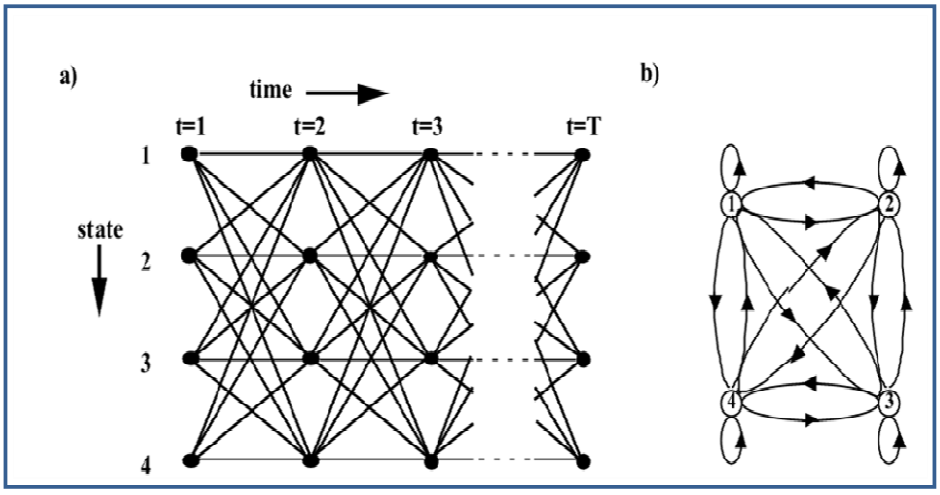

Figure 2.5 (a) Trellis Diagram (b) corresponding graph of 4 states HMM

Mathematically Hidden Markov Model contains five elements.

I. Internal States: These states are hidden and give the flexibility to model different applications. Although they are hidden, usually there is some kind of relation between the physical significance to hidden states.

II. Output: O = {O1, O2, O3, . . ., On} an output observation alphabet.

III. Transition Probability Distribution: A = a_{ij} is a matrix. The matrix defines what the probability to transition from one state to another is.

IV. Output Observation: Probability Distribution B = b_i (k) is probability of generating observation symbol O(k) while entering to state *i* is entered.

V. The initial state distribution (π = {πi}) is the distribution of states before jumping into any state.

Here all three symbols represents probability distributions i.e. A, B and π. The probability distributions A, B and π are usually written in HMM as a compact form denoted by lambda as λ = (A, B, π) [17].

❖ **Related work in the Statistical Algorithm**

Rajib et. al. [18] have proposed Hidden Markov Model based system for English Handwritten character recognition. They have employed global as well as local feature extraction methods. HMM is trained using these feature and experiment is carried out. They have created a database of 13000 samples collected from 100 writers written five times for each character. 2600 samples have been used to train HMM and the rest are used to test recognition model. The recognition rate is achieved 98.26% using proposed system.

Pritpal Singh and Sumit Budhiraja [19] have proposed K Nearest Neighbour algorithm to recognise handwritten Gurumukhi script. They calculate the Euclidean distance between the test point and all the reference points in order to find K nearest neighbours, and then arrange the distances in ascending order

and take the reference points corresponding to the k smallest Euclidean distances. A test sample is then attributed the same class label as the label of the majority of its K nearest (reference) neighbours. The recognition rate is achieved by proposed system is 72.54%.

N. Suguna and Dr. K. Thanushkodi [20] have proposed an improved K Nearest Neighbour algorithm. Genetic Algorithm (GA) is combined with KNN to improve its classification performance. Instead of considering all the training samples and taking k-neighbours, the GA is employed to take k-neighbours straight away and then calculate the distance to classify the test samples. Before classification, initially the reduced feature set is received from a novel method based on Rough set theory hybrid with Bee Colony Optimization (BCO).

Emanuel Indermuhle, Marcus Liwicki and Horst Bunke [21] have proposed Hidden Markov Model to recognise handwritten manuscripts by writers of the 20th century.They proposed two approaches of Hidden Markov Model for training the recognition system. The first approach consists in training the recognizer directly from scratch, while the second adapts it from a recognizer previously trained on a large general off-line handwriting database.

B.V.Dhandra, Gururaj Mukarambi and Mallikarjun Hangarge [22] have proposed handwritten english uppercase character recognition system. Handwriiten images are normalized into 32×32 dimensions. Then the normalized images are divided into 64 zones and their pixel densities are calculated, generating a total of 64 features. These 64 features are submitted to KNN and SVM classifiers with 2 fold cross validation for recognition of the said characters. The recognition accuracy of 97.51% for KNN and 98.26% for SVM classifiers are obtained in case of handwritten English uppercase alphabets.

In [23] k-nearest neighbor classifier approach has been used for the Gujarati character recognition. It used the k-nearest samples to test sample and identifies it to that class which has the largest number of votes. The nearest neighbor is found by using the Euclidean distance measure. For 1-NN classifier the best recognition rate achieved was 67% in the binary feature space and in regular moment space the rate was 48% [24].

Noha A. Yousri, Mohamed S. Kamel and Mohamed A. Ismail have [25] proposed distance based clustering that identifies the natural shapes of clusters. In case of character recognition, where it is natural to have different writing styles for same character, the algorithm can be used to discover the continuity between character feature vectors, which can't be discovered by traditional algorithms.

2.2.3 Structural/Syntactic Algorithm

The recursive description of a complex pattern in terms of simpler patterns based on the shape of the object was the initial idea behind the creation of structural pattern recognition [26].

Structural algorihtm classifies the input patterns on the basis of components of the characters and the relationship among these components. Firstly the primitives of the character are identified and then strings of the primitives are checked on the basis of pre-decided rules [11]. Structural pattern recognition is intuitively appealing because in addition to classification, this approach also provides a description of how the given path constructed from the primitives [27]. Generally a character is represented as a production rules structure, whose left-hand side represents character labels and whose right-hand side represents string of primitives. The right-hand side of rules is compared to the string of primitives extracted from a word. So classifying a character means finding a path to a leaf [11].

❖ **Related work in the Structural Algorithm**

Jonathan J. HULL, Alan COMMIKE and Tin-Kam HO [28] have presented Structural analysis algorithm. The structural analysis algorithm has been fully implemented and tested in the proposed work.

Hewavitharana, S, and H.C. Fernando [29] have proposed a system to recognize handwritten Tamil characters using a two-stage classification approach, for a subset of the Tamil alphabet, which is a hybrid of structural and statistical techniques. In the first stage, an unknown character is pre-classified into one of the three groups: core, ascending and descending characters. Structural properties of the text line are used for this classification. Then, in the second stage, members of the pre-classified group are further analysed using a statistical classifier for final recognition [30].

2.2.4 Neural Network Algorithm

An Artificial Neural Network (ANN) is an information processing paradigm that is inspired by the way biological nervous systems, such as the brain, process information. It is composed of a large number of highly interconnected processing elements (neurons) working in unison to solve specific problems [31]. A neural network is a powerful data modeling tool that is able to capture and represent complex input/output relationships. The motivation for the development of neural network technology stemmed from the desire to develop an artificial system that could perform "intelligent" tasks similar to those performed by the human brain.

Neural Network consists three layers - input layer, hidden layer (optional) and output layer.

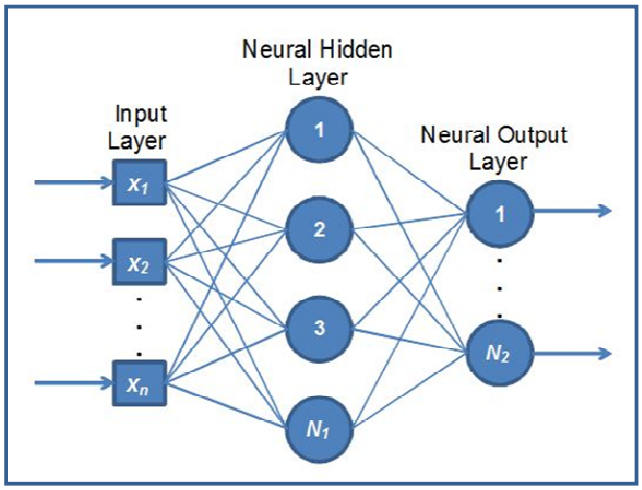

Figure 2.6 Neural Network Architecture

I. Input Layer

The input layer is the conduit through which the external environment presents a pattern to the neural network. Input layer take input from the external world and encode it into a convenient form. Every input neuron should represent some independent variable that has an influence over the output of the neural network.

The number of input neurons for the OCR is the number of pixels that might represent any given character. A character which represents by a 5×7 grids has 35 pixels. So it has 35 input neurons.

II. Hidden Layer

Hidden layer can't see or act upon the outside world directly. These inter-neurons communicate only with other neurons. Deciding the number of neurons in the hidden layers is a very important part of deciding your overall neural network architecture. Both the number of hidden layers and the number of neurons in each of these hidden layers must be carefully considered [32].

There are many rule-of-thumb methods for determining the correct number of neurons to use in the hidden layers, such as the following:

- The number of hidden neurons should be between the size of the input layer and the size of the output layer.
- The number of hidden neurons should be 2/3 the size of the input layer, plus the size of the output layer.
- The number of hidden neurons should be less than twice the size of the input layer.

III. Output Layer

The output layer of the neural network is what actually presents a pattern to the external environment. The number of output neurons should be directly related to the type of work that the neural network is to perform.

The number of output neurons used by the OCR program will vary depending on how many characters the program has been trained to recognize. The default training file that is provided with the OCR program is used to train it to recognize 26 characters. Using this file, the neural network will have 26 output neurons. Presenting a pattern to the input neurons will fire the appropriate output neuron that corresponds to the letter that the input pattern represents [33].

In a neural network, each node performs some simple computations, and each connection conveys a signal from one node to another, labeled by a number called the “connection strength” or “weight” indicating the extent to which a signal is amplified or diminished by a connection.

2.2.4.1 Types of Neural Network

Neural network architectures can be classified as, Feed Forward network, Feedback (Recurrent) networks and Self Organizing Map.

2.2.4.1.1 Feed Forward Neural Network

In feed forward neural network, the information moves in only one direction, forward, from the input nodes, through the hidden nodes (if any) and to the output nodes. There are no cycles or loops in the network [34]. There are two types of Feed Forward Network. Single Layer Feed Forward Network (An input layer directly connected to output layer) and Multi-Layer Feed Forward Network (an input layer and an output layer with one or more hidden layers).

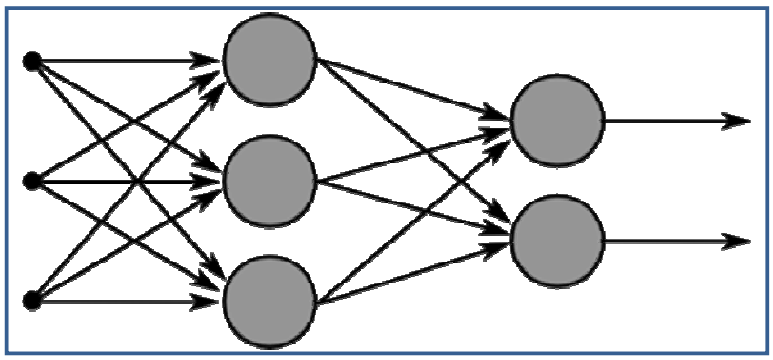

Figure 2.7 Feed Forward Neural Network

2.2.4.1.2 Feedback(Recurrent) Neural Network

A Feed forward neural network having one or more hidden layers with at least one feedback loop is known as feedback network. A feedback neural network is any network whose neurons send feedback signals to each other. The feedback may be a self-feedback, i.e., where output of neuron is feedback to its own input [33]. Unlike feed forward neural networks, feedback network can use their internal memory to process arbitrary sequences of inputs. This makes them applicable to tasks such as unsegment connected handwriting recognition, where they have achieved the best known results [35].

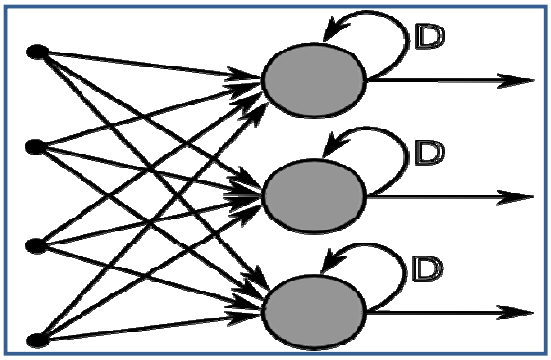

Figure 2.8 Feed Back Neural Network

2.2.4.1.3 Self-Organizing Map

A self-organizing map (SOM) or self-organizing feature map (SOFM) is a type of artificial neural network (ANN) that is trained using unsupervised learning to produce a low-dimensional (typically two-dimensional), discretized representation of the input space of the training samples, called a map. Self-organizing maps are different from other artificial neural networks in the sense that they use a neighborhood function to preserve the topological properties of the input space [36]. The Self Organizing Map (SOM) is invented by Teuvo Kohonen. So it is also called Kohonen Self Organizing Map. 2-dimensional Kohonen layer is shown in Figure 2.9.

Like most artificial neural networks, SOMs operate in two modes: training and mapping. "Training" builds the map using input examples (a competitive process, also called vector quantization), while "mapping" automatically classifies a new input vector. The goal of learning in the self-organizing map is to cause different parts of the network to respond similarly to certain input patterns [36].

The network (the units in the grid) is initialized with small random values. A neighborhood radius is set to a large value. The input is presented and the Euclidean distance between the input and each output node is calculated. The

node with the minimum distance is selected, and this node, together with its neighbors within the neighborhood radius, will have their weights modified to increase similarity to the input. The algorithm results in a network where groups of nodes respond to each class thus creating a map of the found classes [37].

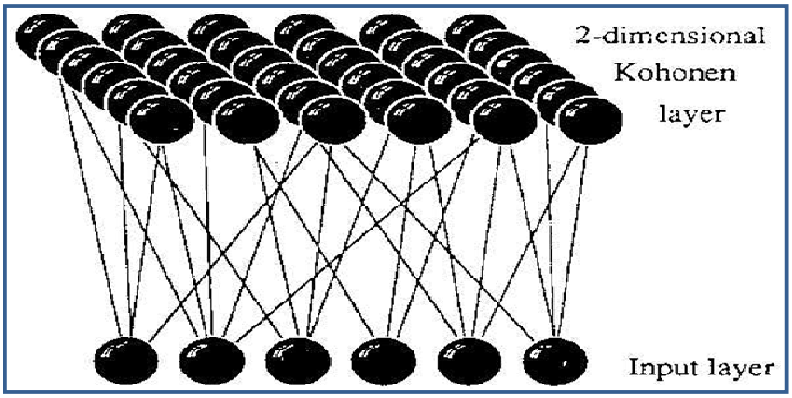

Figure 2.9 Two Dimensional Kohonen Layer

2.2.4.2 Training of Neural Network

Supervised training and unsupervised training are the types of training neural network.

2.2.4.2.1 Supervised Training

Supervised training requires the pairing of each input vector with a target vector representing the desired output; together these are called a training pair. Usually a network is trained over a number of such training pairs. An input vector is applied, the output of the network is calculated and compared to the corresponding target vector, and the difference (error) is fed back through the network and weights are changed according to algorithms that tend to minimize the error. The vectors of the training sets are applied sequentially, and errors are calculated and weight adjusted for each vector, until the error for the training set is at an acceptably low level[38] [39, 40].

2.2.4.2.2 Unsupervised Training

It requires no target vector for the outputs, and hence, no comparisons to pre-determined ideal response. The training set consists of input vectors. The training algorithm modifies network weights to produce output vectors that are consistent; that is, both applications of one of the training vectors or application of vector that is sufficiently similar to it will produce the same pattern of outputs. The training process, therefore, extracts the statistical properties of the training set and group's similar vector into classes. Applying a vector from a given class to the input will produce a specific output vector, but there is no way to determine prior to training which specific output pattern will be produce by given input vector class. Hence the outputs of such a network must generally be transformed into a comprehensible form subsequent to the training process [38] [39, 40].

- **Multilayer Perceptron**

A multilayer perceptron (MLP) is a feed forward artificial neural network model that maps sets of input data onto a set of appropriate outputs. A Multilayer perceptron consists of multiple layers of nodes in a directed graph, with each layer fully connected to the next one. Except for the input nodes, each node is a neuron (or processing element) with a nonlinear activation function. MLP utilizes a supervised learning technique called back propagation for training the network [41] [42] [43]. MLP is a modification of the standard linear perceptron and can distinguish data that are not linearly separable [41] [44].

The goal of this type of network is to create a model that correctly maps the input to the output using historical data so that the model can then be used to produce the output when the desired output is unknown.

❖ **Back Propagation Training Algorithm**

Back propagation is a simplest and most general methods used for supervised training of multilayered neural networks. The back-propagation algorithm uses supervised learning, which means that we provide the algorithm with examples of the input and outputs we want the network to compute, and then the error is computed. Back-propagation algorithm is the most familiar, powerful, and effective algorithm used to train the multilayer perception (MLP) networks.

The back propagation algorithm trains a given feed-forward multilayer neural network for a given set of input patterns with known classifications. When each entry of the sample set is presented to the network, the network examines its output response to the sample input pattern. The output response is then compared to the known and desired output and the error value is calculated. Based on the error, the connection weights are adjusted [45]. The aim of the back-propagation algorithm is to reduce the error, until the ANN learns the training data. The training starts with random weights, and aim is to adjust them to arrive at minimal error.

The back propagation learning algorithm can be divided into two phases: propagation and weight update.

Phase 1: Propagation

Each propagation involves the following steps:

I. Forward propagation of a training pattern's input through the neural network in order to generate the propagation's output activations.

II. Backward propagation of the propagation's output activations through the neural network using the training pattern target in order to generate the deltas of all output and hidden neurons.

Phase 2: Weight Update

For each weight-synapse follow the following steps:

I. Multiply its output delta and input activation to get the gradient of the weight.

II. Subtract a ratio (percentage) of the gradient from the weight.

This ratio (percentage) influences the speed and quality of learning; it is called the learning rate. The greater the ratio, the faster the neuron trains; the lower the ratio, the more accurate the training is. The sign of the gradient of a weight indicates where the error is increasing; this is why the weight must be updated in the opposite direction [46].

Repeat phase 1 and 2 until the performance of the network is satisfactory.

❖ **Related work in the Neural Network Algorithm**

Parveen Kumar, Nitin Sharma and Arun Rana [47] have proposed handwritten character recognition system using multi-layer feed forward back propagation neural network without feature extraction. For the neural network, each character is resized into 70x50 pixels, which is directly subjected to training. That is, each resized character has 3500 pixels and these pixels are taken as features for training the neural network. The proposed system has 4 layers – one input layer, one output layer and two hidden layers, having 200 neurons in the first hidden layer and 100 neurons in the second hidden layer. The recognition rate is 80.96%.

Yusuf Perwej and Ashish Chaturvedi [48] have proposed neural networks for developing a system that can recognize handwritten English alphabets. Each English alphabet is represented by binary values that are used as input to a simple feature extraction system, whose output is fed to neural network system. The recognition rate is 82.5%.

Sameeksha Barve [49] has proposed optical character recognition based on artificial neural network (ANN). The ANN is trained using the Back Propagation algorithm. In the proposed system, each typed English letter is represented by binary numbers that are used as input to a simple feature extraction system whose output, in addition to the input, are fed to an ANN. Afterwards, the Feed Forward Algorithm gives insight into the enter workings of a neural network followed by the Back Propagation Algorithm which compromises Training, Calculating Error, and Modifying Weights.

MdFazlul Kader and Kaushik Deb [50] have proposed artificial neural network based simple colour and size invariant character recognition system using feed-forward neural network to recognize English alphanumeric characters. They use single layer feed forward neural network so there is no hidden layer and only one type of weight, input-output weight. The input character matrix is normalized into 12×8 matrix for size invariant recognition and fed into the proposed network which consists of 96 input and 36 output neurons. They have tested network by more than 20 samples per character on average and give 99.99% accuracy only for numeric digits (0-9), 98% accuracy only for letters (A-Z) and more than 94% accuracy for alphanumeric characters by considering inter-class similarity measurement.

Pradeep et. al. [51] have proposed neural network based classification of handwritten character recognition system. Each individual character is resized to 30 X 20 pixels for processing. They are using binary features to train neural network. However such features are not robust. In post processing stage, recognized characters are converted to ASCII format. Input layer has 600 neurons equal to number of pixels. Output layer has 26 neurons as English has 26 alphabets. Proposed ANN uses back propagation algorithm with momentum and adaptive learning rate [52].

Velappa et.al. [53] have proposed multi scale neural network based approach. Proposed system first convert camera captured RGB image to binary image. Width to Height Ratio (WH), Relative Height (RH) ratio, Relative Width ratio (RW) is calculated to remove unnecessary connected components from image. For multi scale neural network, detected character is resized to 20 X 28 pixels, 10 X 14 pixels and 5 X 7 pixels. Binary features of these different resolution images are given to three layer feed forward back propagation algorithm [52].

Akash Ali et al. [54] have proposed handwritten Bangla character recognition using Back propagation Feed-forward neural network. First, Create binary image then, extract the feature and form input vector. Then, apply the input vector in the neural network. The experimental result shows that the proposed recognition method gives 84% accuracy and less computational cost than other method.

V. Kalaichelvi and Ahammed Shamir Ali [55] have proposed Application of Neural Networks in Character Recognition. The application of neural networks in recognizing characters from a printed script is explored. Contrast to traditional methods of generalizing the character set, a highly specific character set is trained for each type.

Ujjwal Bhattacharya, B. B. Chaudhuri [56] used a distinct MLP classifier. They worked on Devanagari, Bengali and English handwritten numerals. A back propagation (BP) algorithm was used for training the MLP classifiers. It provided 99.27% and 99.04% recognition accuracies on the original training and test sets of Devanagari numeral database, respectively [57].

2.2.5 Support Vector Machine

Support vector machines (SVMs also support vector networks) are a set of related supervised learning methods used for classification.

SVMs are relatively new approach compared to other supervised classification algorithms, they are based on statistical learning theory developed by the Russian scientist Vladimir Naumovich Vapnik back in 1962 and since then, his original ideas have been perfected by a series of new techniques and algorithms [58].

Support vector machines have proved to achieve good generalization performance with no prior knowledge of the data. The principle of an SVM is to map the input data onto a higher dimensional feature space nonlinearly related to the input space and determine a separating hyper plane with maximum margin between the two classes in the feature space[60] [61]. This approach, in general, guarantees that the larger the margin is the lower is the generalization error of the classifier [62].

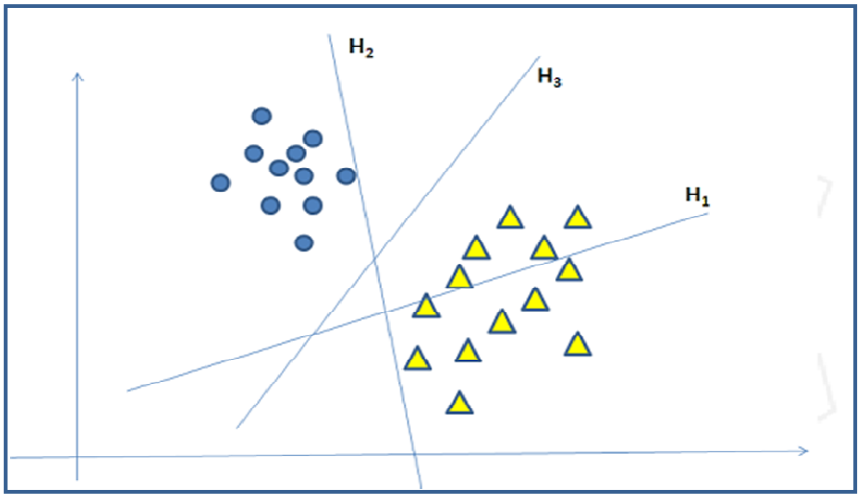

Figure 2.10 Separation hyper planes. H1does not separate the two classes; H2separates but with a very tinny margin between the classes and H3separates the two classes with much better margin than H2

If such hyper plane exists, it is clear that it provides the best separation border between the two classes and it is known as the maximum-margin hyper plane and such a linear classifier is known as the maximum margin classifier [62].

A support vector machine is a maximal margin hyper plane in feature space built by using a kernel function in gene space. This results in a nonlinear boundary in the input space. The optimal separating hyper plane can be determined without any computations in the higher dimensional feature space by using kernel functions in the input space [60]. Commonly used kernels include:-

I. Linear Kernel:

$$K(x, y) = x * y$$

II. Radial Basis Function (Gaussian) Kernel:

$$K(x, y) = \exp(-\|x - y\|^2/2\sigma^2)$$

III. Polynomial Kernel:

$$K(x, y) = (x * y + 1)^{d}$$

For multi-class classification, binary SVMs are combined in either one against-All or one-against-one (pair wise) scheme.

I. One against All

The "one against all" strategy consists of constructing one SVM per class to separate members of that class from members of other classes. Usually, classification of an unknown pattern is done according to the maximum output among all SVMs.

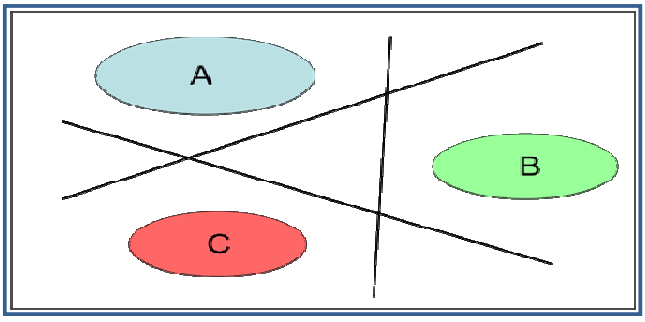

Figure 2.11 Diagram of binary One against all region boundaries on a basic problem

II. One against One

The “one against one” strategy, also known as “pair wise coupling”, “all pairs” or “round robin”, consists in constructing one SVM for each pair of classes. Thus, for a problem with c classes, c(c-1)/2 SVMs are trained to distinguish the samples of one class from the samples of another class. Usually, classification of an unknown pattern is done according to the maximum voting, where each SVM votes for one class [63].

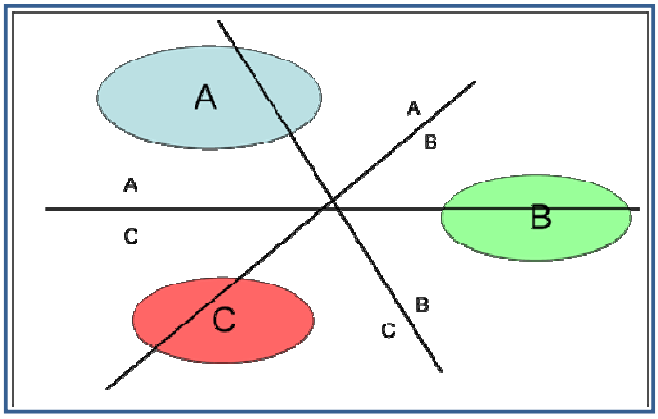

Figure 2.12 Diagram of binary One against One region boundaries on a basic problem

SVM algorithm is robust, accurate and very effective even in cases where the number of training samples are small.

❖ **Related Work in Support Vector Machine**

Nasien et al. [64] have proposed a recognition model for English handwritten (lowercase, uppercase and letter) character recognition that uses Freeman chain code (FCC) as the representation technique of an image character. Support vector machine (SVM) has been chosen for the classification. The proposed recognition model, built from SVM classifiers was efficient enough to show that applying the proposed model, a relatively higher accuracy of 98.7% for the problem of English handwritten recognition was reached [65].

Shailedra Kumar Shrivastava and Sanjay S. Gharde [66] have proposed Support Vector Machine for Handwritten Devanagari Numeral Recognition. Binary classification techniques of Support Vector Machine is implemented and linear kernel function is used in SVM. This linear SVM produces 99.48% overall recognition rate.

Munish Kumar, M. K. Jindal and R. K. Sharma [67] have proposed Support Vector Machine for Handwritten Gurumukhi characters. The classifier that has been employed is SVM with three flavors, i.e., SVM with linear kernel, SVM with polynomial kernel and SVM with RBF kernel. The features have been inputted to the classifiers individually and have also been inputted simultaneously. The recognition rate achieved by proposed system is 94.29%.

Parveen Kumar, Nitin Sharma and Arun Rana [68] have proposed handwritten character recognition system using SVM. For the SVM classifier, recognition model is divided in two phases namely, training and testing phase. In the training phase 25 features are extracted from each character and these features are used to train the SVM. In the testing phase SVM classifier is used to recognize the characters. Recognition rate using linear function is 94.8%.

Anshuman Sharma [57] has proposed handwritten digit Recognition using Support Vector Machine. In this proposed work the SVM (binary classifier) is applied to multi class numeral recognition problem by using one-versus-rest type method. The SVM is trained with the training samples using linear kernel.

Pritpal Singh and Sumit Budhiraja [19] have proposed SVM algorithm to recognise handwritten Gurumukhi script. They use Radial Basis Function (Gaussian) Kernel and Polynomial Kernel and achieved 73.02% and 95.04% recognition rate respectively.

ZHAO Bin et. al. [69] have used Support Vector Machine for classification and its Application in Chinese check recognition system. The experiment on NIST numeral database and the actual check samples shows that comparison with other classifiers, SVM possesses better generalization ability. They have got 99.95% accuracy [70].

Gita Sinha et.al. [70] have proposed SVM for Handwritten Gurumukhi Character Recognition. In the proposed system Radial Basis Function kernel is used and they achieved 95.11% recognition rate.

N. Shanthi and K. Duraiswamy [71] have proposed a recognition system for offline unconstrained handwritten Tamil characters based on support vector machine. Due to the difficulty in great variation among handwritten characters, the system is trained with 106 characters and tested for 34 selected Tamil characters. The characters are chosen such that the sample data set represents almost all the characters. Pixel densities are calculated for different zones of the image and these values are used as the features of a character. These features are used to train and test the support vector machine [72].

In [73] authors propose the Support Vector Machine (SVM) based recognition scheme towards the recognition of Gujarati handwritten numerals. A technique based on affine invariant moments for feature extraction is applied and the recognition rate of 91% approximately [74].

2.2.6 Decision Tree Classifier

The concept of decision tree is decomposition of complex problem into smaller, more manageable whereby it represents the relationship among attribute and decision in a diagram that mimic to tree [75] [76]. The classification is produced by algorithm that identifies various ways of splitting a data into branch-like tree segment. The segmentations basically comprise 3 structures; internal node donates a test on attribute, branch node represent an

outcome of the test, and leaf nodes represent class distribution. Decision Tree has several algorithms such as ID3, C4.5 (extension of ID3), and CART. Generally, tree induction and tree pruning are two main processes in Decision tree classifier. Tree induction is an iterative training process which involves splitting the attributes into smaller subsets. This process starts by analyzing the whole dataset to find the condition attributes whereby when it is selected as a splitting rule, it will result in nodes that are most different from each other with respect to the target class. Subsequently, the tree will be generalized in pruning process by removing least reliable tree branches and accuracy will be improved [75].

A Decision Tree is a tree in which each branch node represents a choice between a numbers of alternatives, and each leaf node represents a classification or decision [75]. In the decision tree, the root and internal nodes contain attribute test conditions to separate records that have different characteristics.

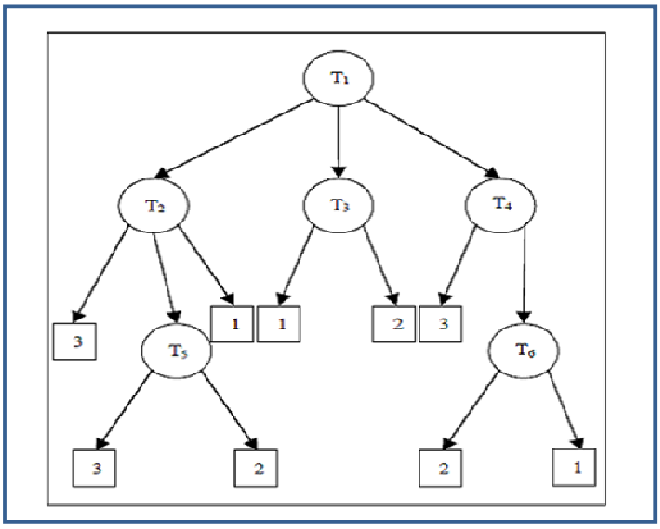

Figure 2.13 Decision Tree

The design of a decision tree classifier can be decomposed into following tasks:

I. The appropriate choice of tree structure.

II. The choice of feature subsets to be used at each internal node.

III. The choice of the decision rule or strategy to be used at each internal node.

❖ **Related Work in Decision Tree Classifier**

N. Shobha Rani and Smitha Madhukar [77] have proposed a decision tree based font style/size independent Kannada printed character recognition system. They have provided improved accuracy in recognizing the complex or overlapping characters and proved to be efficient by obtaining around 97% - 99% of accuracy.

Zhang Ping and Chen Lihui [78] have proposed a hybrid classification system with neural network and decision tree for handwritten numeral recognition.

A. Amin and S. Singh [79] have proposed recognition of hand printed Chinese Characters using Decision Tree. The proposed system was tested with 900 characters written by different writers from poor to acceptable quality and rate of recognition obtained was 84%.

2.3 Optical Character Recognition Tools

Optical Character Recognition tools are used to convert handwritten or typewritten characters to machine editable form. OCR tools convert a scanned image or PDF file to text file. OCR tools are either front-end or backend tool for software. Today many types of OCR software available like: Web OCR (Online OCR), Desktop OCR etc.

Web OCR is also known as Online OCR. Web OCR will require that you upload your files on the internet to their servers, so there may be privacy

concerns as well as time/bandwidth concerns if your document is big. Most have limits to file size and count of pages to process daily/weekly that they will process for free; for bigger jobs they require to buy extra processing power. On the flip side, many of these services are really good at the OCR itself [80].

Desktop OCR is also known as Offline OCR. With Desktop OCR you don't need to worry about uploading sensitive information to foreign servers, or whether your file will take too long to upload. Some desktop OCR programs generally give better text review options, and some have scanning functionality integrated [80].

Researcher has studied following tools and software. Among them some are open source and free.

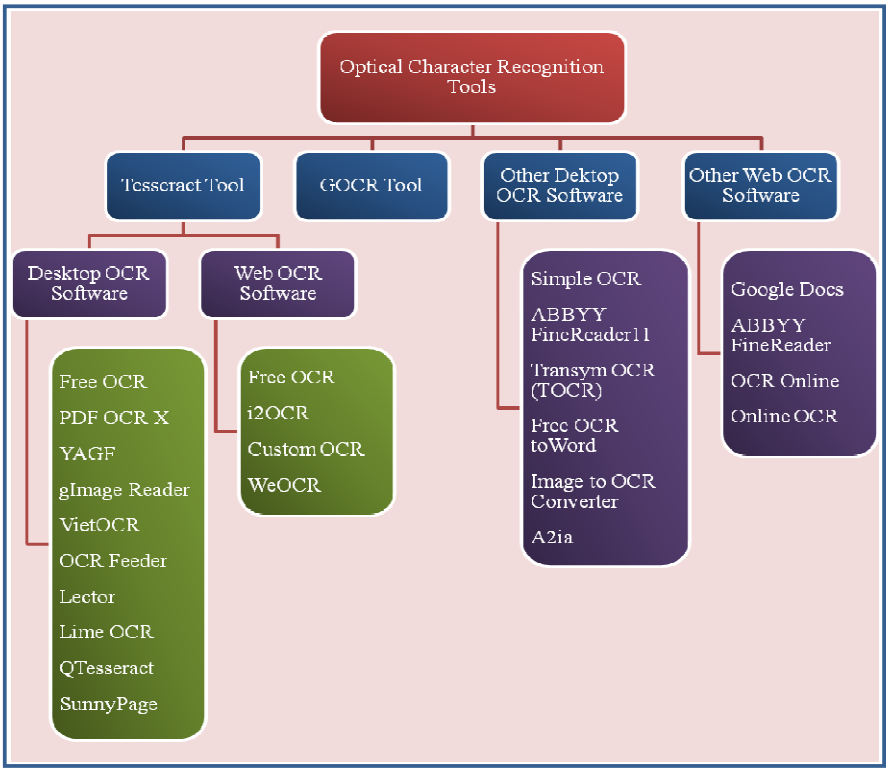

Figure 2.14 Optical character Recognition Tools

2.3.1 Tesseract Tool

Tesseract is free software, released under the Apache License and development has been sponsored by Google since 2006.Tesseract is one of the most accurate open source OCR engines currently available.

❖ **History of Tesseract**

The Tesseract engine was originally developed as proprietary software at Hewlett Packard labs in Bristol, England and Greeley, Colorado between 1985 and 1994, with some more changes made in 1996 to port to Windows, and some migration from C to C++ in 1998. A lot of the code was written in C, and then some more was written in C++. Since then all the code has been converted to at least compile with a C++ compiler. Very little work was done in the following decade. It was then released as open source in 2005 by Hewlett Packard and the University of Nevada, Las Vegas (UNLV). Tesseract development has been sponsored by Google since 2006 [81].

❖ **Features**

Tesseract was in the top three OCR engines in terms of character accuracy in 1995. It is available for Linux, Windows and Mac OS X [81].

Tesseract up to and including version 2 could only accept TIFF images of simple one column text as inputs. These early versions did not include layout analysis and so inputting multi-columned text, images, or equations produced a garbled output. Since version 3.00 Tesseract has supported output text formatting, OCR positional information and page layout analysis. Support for a number of new image formats was added using the Leptonica library. Tesseract can detect whether text is moonscape or proportional [81].

The initial versions of Tesseract could only recognize English language text. Starting with version 2 Tesseract was able to process English, French, Italian, German, Spanish, Brazilian Portuguese and Dutch. Starting with version 3 it

can recognize Arabic, English, Bulgarian, Catalan, Czech, Chinese (Simplified and Traditional), Danish, German (standard and Fraktur script), Greek, Finnish, French, Hebrew, Hindi, Croatian, Hungarian, Indonesian, Italian, Japanese, Korean, Latvian, Lithuanian, Dutch, Norwegian, Polish, Portuguese, Romanian, Russian, Slovak (standard and Fraktur script), Slovenian, Spanish, Serbian, Swedish, Tagalog, Tamil, Thai, Turkish, Ukrainian and Vietnamese. Tesseract can be trained to work in other languages too [81].

Tesseract includes the English training data. If you want to use another language, download the appropriate training data, unpack it using 7-zip, and copy the .traineddata file into the 'tessdata' directory, probably C:\Program Files\Tesseract OCR\tessdata.

Tesseract's output will be very poor quality if the input images are not preprocessed to suit it: Images (especially screenshots) must be scaled up such that the text x-height is at least 20 pixels, any rotation or skew must be corrected or no text will be recognized, low-frequency changes in brightness must be high-pass filtered, or Tesseract's binarization stage will destroy much of the page, and dark borders must be manually removed, or they will be misinterpreted as characters [81].

Tesseract does not come with a GUI and is instead run from the command-line interface [82]. Tesseract version 3.03 is released and available for use.

2.3.1.1 Desktop OCR Software using Tesseract Tool

Tesseract tool is used as a backend for Desktop OCR Software. Researcher has studied following Desktop OCR software of Tesseract Tool.

1. Free OCR
2. PDF OCR X
3. YAGF
4. GimageReader

5. VietOCR
6. OCRFeeder
7. Lector
8. Lime OCR

Above software are used as frontend for Tesseract Tool.

1. Free OCR

Free OCR is a useful tool for scanning and extracting text from images and PDFs. Free OCR supports most image files and multi-page TIFF files. It can handle PDF formats and is also compatible with TWAIN devices like scanners. Free OCR is a complete scan OCR program [83].

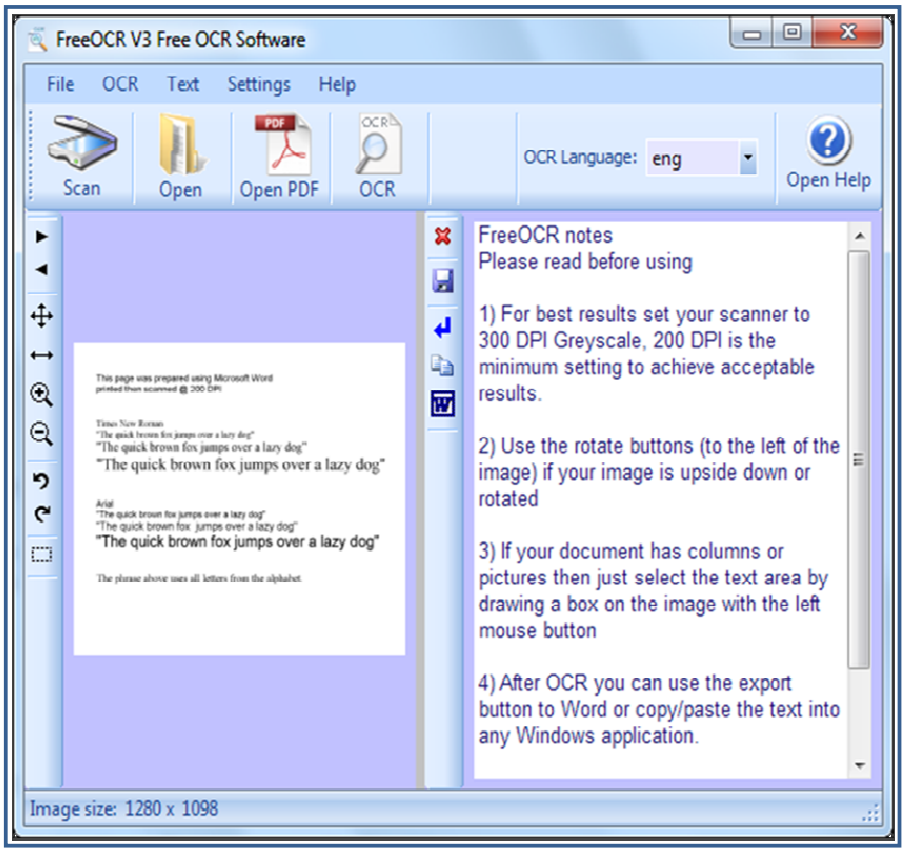

Figure 2.15 Screenshot of Free OCR Software

Before starting the one click conversion process, you can adjust the image contrast for better readability [84]. Additional languages also can be installed in Free OCR. For that, download language file from code.google.com website.

Free OCR can't recognize formatting. Scanning a page has to do a lot with resolutions, contrasts and clarity of fonts.

❖ **How it works?**

I. Open - Open an image file.

II. Open PDF – Open a PDF file.

III. OCR - Convert an image or PDF file to text file.

2. PDF OCR X

PDF OCR X is a simple drag-and-drop utility for Mac OS X and Windows that converts PDFs and images into text documents or searchable PDF files. It uses advanced OCR (optical character recognition) technology to extract the text of the PDF even if that text is contained in an image. This is particularly useful for dealing with PDFs that were created via a Scan-to-PDF function in a scanner or photo copier [85]. It supports over 60 languages. It also supports batch processing.

❖ **How it works?**

I. Select file - Select an image or PDF file.

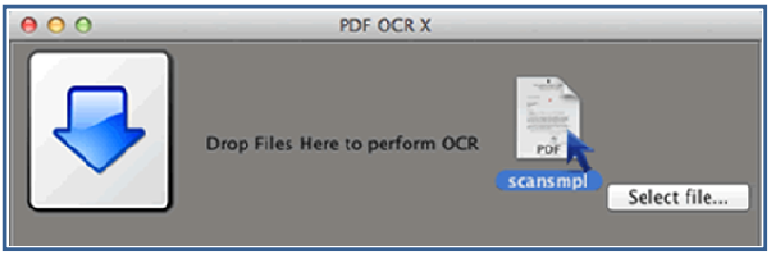

Figure 2.16 Screenshot of PDF OCR X

II. Select OCR conversion settings.

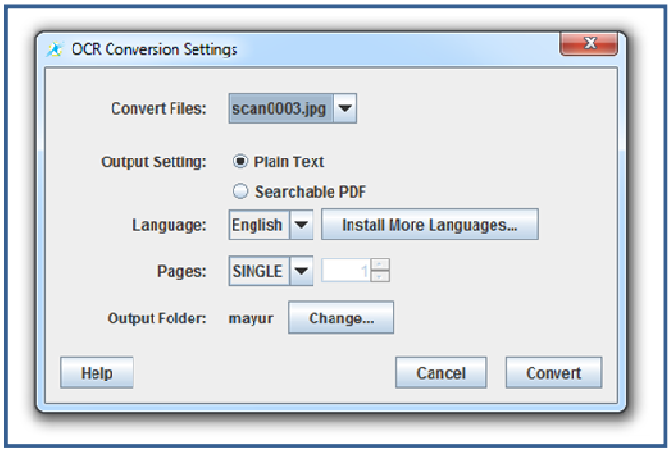

Figure 2.17 Conversion Settings of PDF OCR X

III. Convert- Converts PDF or image to text or searchable PDF depending on your conversion settings.

3. YAGF

YAGF is a graphical interface for cuneiform and tesseract text recognition tools on the Linux platform. With YAGF you can scan images via XSane, import pages from PDF documents, perform image preprocessing and recognize texts using cuneiform from a single command center. YAGF also makes it easy to scan and recognize several images sequentially [86].

It provides rotation and automatic skew correction facility. User can manually define and adjust recognition regions. YAGF also provides some facilities for a multi-page recognition.

4. gImageReader

GImageReader reads text from images and PDFs. It can import images from disk, scanning devices, clipboard and screenshots. User can manually define and adjust recognition regions. It supports multipage PDF documents. It

supports Spellchecking for output text (if corresponding dictionary installed). It removes line breaks in output text [87].

5. VietOCR

VietOCR, available in Java and .NET executable, is a GUI frontend for Tesseract OCR engine. Both versions support similar graphic user interface and are capable of recognizing text from images of common formats. The program can also function as a console application, executing from the command line. Language data for Vietnamese and English is already bundled with the program. Data for other languages can be downloaded from Tesseract website and should be placed into tessdata folder [88].

VietOCR runs on multi-platform (Java version only) i.e. Windows, Solaris, Linux/Unix, Mac OS X and Others. It extracts text from PDF, TIFF, JPEG, GIF, PNG, BMP image formats. It supports batch processing and scanning facility.

6. OCRFeeder

OCRFeeder is a document layout analysis and optical character recognition system. It automatically outlines its contents, distinguish between what's graphics and text and perform OCR over the latter. It generates multiple formats being its main one ODT [89].

It is a complete GTK graphical user interface that allows the users to correct any unrecognized characters, defined or correct bounding boxes, set paragraph styles, clean the input images, import PDFs, save and load the project, export everything to multiple formats, etc. [89].

7. Lector

Lector is a graphical OCR solution for GNU/Linux based on Python, Qt4 and tesseract OCR. Lector lets you select areas on which you want to do OCR.

Then you can run tesseract-ocr simply clicking a button. The resulting text can be proofread, formatted and edited directly in Lector [90]. It supports rotation of images, spellchecking, text editing. It exports output to ODT (by default), TXT, HTML or PDF.

8. Lime OCR

Lime OCR is built with tessearact-ocr. Lime OCR was initially developed for internal use of Lime Consultants, and now published under GNU General Public License v3. Lime OCR is free, simple to use and currently supports 29 languages, and supports all tesseract-ocr trained data files [91]. Lime OCR supports over 50 types of images and PDF. It supports rotation and cropping.

9. QTesseract

QTesseract is a Graphical User Interface for the Tesseract OCR. The interface allows the user to load, recognize and save result.

10. SunnyPage

SunnyPage OCR is a GUI frontend for Tesseract OCR engine with automatic adjustment of image brightness; image processing and PDF support [92].

2.3.1.2 Web OCR Software using Tesseract OCR

Tesseract tool is also used as a backend for Web OCR Software. Researcher has studied following Web OCR software of Tesseract Tool.

1. Free OCR
2. I2OCR
3. Custom OCR
4. WeOCR

Above web OCR software are frontend for Tesseract tool which are discussed here.

1. Free OCR

Free-OCR is a free online OCR tool. It takes a JPG, GIF, TIFF, BMP or PDF (only first page). It can handle images with multi-column text and supports many languages. The images must not be larger than 2MB, no wider or higher than 5000 pixels and there is a limit of 10 images uploads per hour [93].

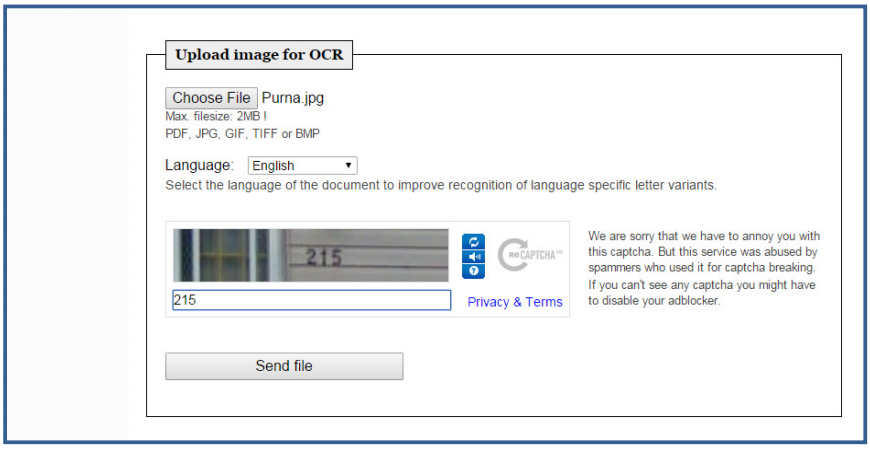

Figure 2.18 Screenshot of Free OCR

2. i2OCR

I2OCR is an online service of OCR. I2OCR reads text from JPG, PNG, BMP, TIFF, PBM, PGM and PPM formats and convert it into text, word and PDF format.

I2OCR enables you to upload an image file from a URL (web, cloud etc.). It supports 64 recognition languages. It analyzes the layout of the document and can extract text from multiple columns [94]. It gives facility to edit extracted text online using Google Docs or translated using Google or Bing translation service. I2OCR displays recognized text as well as input source image side by side to facilitate reviewing misrecognized words. I2OCR cannot read text from PDF File.

Figure 2.19 Screenshot of i2OCR

3. CustomOCR

CustomOCR read text from scanned documents, webcam images of license plates or photo camera images of bank checks [95].

CustomOCR solutions can be installed in two variants, local and cloud-hosted, each of which has its own advantages.

Local Installation: - It allows to install CustomOCR solution locally, to ensure the best performance in time-critical applications and the tightest integration with application components. For local installations a compiled executable (Windows or Linux) file is used, which is a console application. It has very light-weight startup code, so it can be run for every image separately and multiple instances can be run in parallel. Input parameters and image names can be specified via command line or a config file, or taken from a specified directory on the hard drive or from a network location [95].

Cloud-hosted Installation**: -** This type of installation fits well for applications which allow for longer response times, and boasts unlimited scalability in the sense of processing power. In this case, text extraction solution is hosted on our server located in the Amazon EC2 cloud. Application will communicate with the CustomOCR server via the Internet using a Restful protocol, which is in fact no more than a set of ordinary HTTP requests. To send an image, you

will need to wrap it into a POST request and send it to the CustomOCR server. To obtain results, you'll send one or more GET requests and download the files containing your processing results [95].

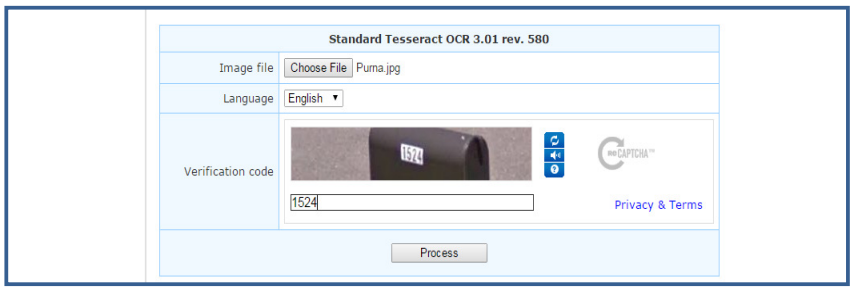

Figure 2.20 Screenshot of CustomOCR

4. WeOCR

WeOCR is a platform for Web-enabled OCR (Optical Character Reader/Recognition) systems that enables people to use character recognition over networks [92]. OCRextrACT is an online access to the tesseract via a WeOCR interface.

Figure 2.21 Screenshot of WeOCR

2.3.2 GOCR Tool

GOCR is again a Command based tool same as Tesseract, it is also used as a backend to other programs. GOCR is an OCR (Optical Character Recognition)

program, developed under the GNU Public License. It converts scanned images to text files. Joerg Schulenburg started the program, and now leads a team of developers [96]. GOCR program is written in C language and it runs on Linux, Windows and OS/2 platform.

It is a simple and fast engine which does not require any training data. Its recognition process takes two passes. In first pass, entire document is called. In second pass the unknown characters are called [97] [98].

GOCR claims it can handle single-column sans-serif fonts of 20–60 pixels in height. It reports trouble with serif fonts, overlapping characters, handwritten text, heterogeneous fonts, noisy images, large angles of skew, and text in anything other than a Latin alphabet. GOCR can also translate barcodes [99] [100].

2.3.3 Other Desktop OCR Software

There are also other desktop OCR software are available. Researcher has studied following Desktop OCR software.

1. Simple OCR
2. ABBYY FineReader 11
3. Transym OCR (TOCR)
4. Free OCR to Word
5. Image to OCR Converter
6. A2ia

Above software are discussed in detail.

1. Simple OCR

Simple OCR is desktop Software. It is a proprietary optical character recognition application developed originally by Cyril Cambien of France under the title WOCAR (until v2.5 in 2001) [102]. Simple OCR offers

Machine print and Hand written Recognition. It provides unlimited license for machine print engine but for hand written engine only 14 days. Simple OCR uses its own OCR engine that is capable of learning the fonts in a particular document [102]. It is a free for all non-commercial purposes. It converts .TIFF, .JPG and .BMP file into .DOC or .TXT file. It is 32 bit application.

Simple OCR gives facility to interactive correction with suggestions from dictionary. Simple OCR has the ability to capture and retain pictures from the document. It includes both single file and batch of files processing mode [102]. Simple OCR gives facility to extract the text from a certain area in a document; there is no need to OCR an entire document only to use a small portion of it. With Simple OCR, OCR only what you need [103].

Simple OCR cannot convert PDF file into text file. Font and format detection is not possible in simple OCR [103]. Simple OCR handles only bi-level (black & white) and grayscale images.

❖ **How it Works?**

I. As shown in the Figure 2.22, select either Machine Print or Hand Writing. Machine Print is used to convert printed document into text file and Hand Writing is used to convert handwritten document to text file.

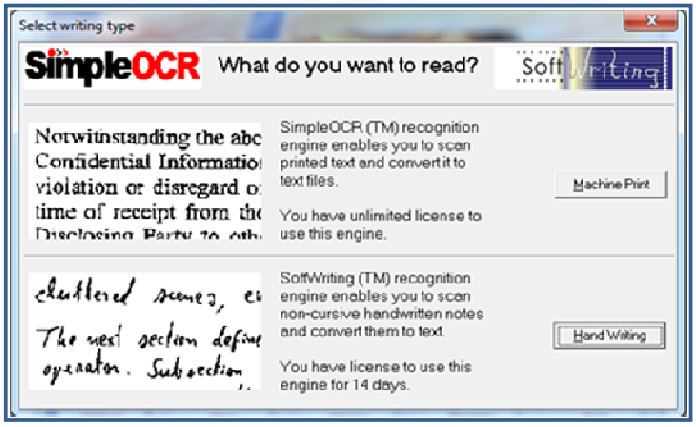

Figure 2.22 Screenshot of Simple OCR

II. Adds Page – Select an image file.

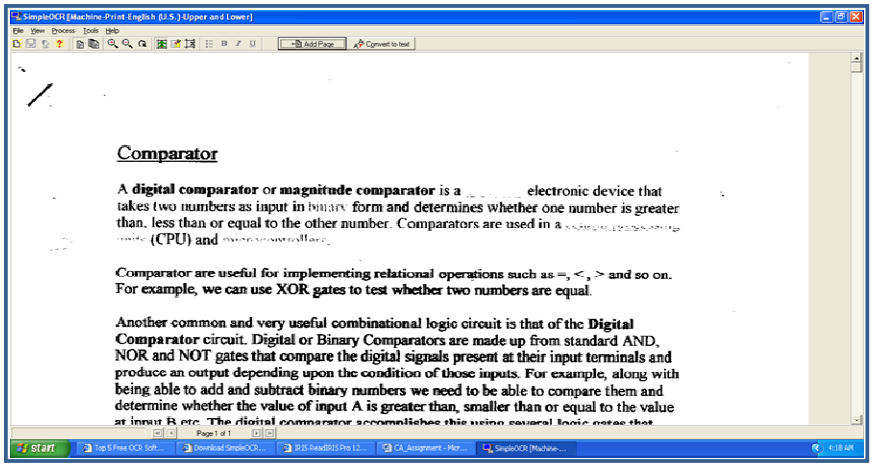

Figure 2.23 Screenshot of Simple OCR Process

III. Convert to text- Perform OCR.

2. ABBYY FineReader 11

ABBYY Fine Reader 11 efficiently converts an image file or PDF file into editable and searchable text file. Fine Reader creates e-books from scans of paper documents, PDFs and digital photographs.

ABBYY Fine Reader Engine is an OCR SDK that gives developers, integrators and BPOs the tools they require to integrate optical text recognition technologies into their applications. The ABBYY recognition platform delivers award-winning OCR, intelligent character recognition (ICR), barcode, checkmark, field-level/zonal recognition and PDF conversion enabling scanned documents and images to be transformed into searchable and editable document formats.

Fine Reader 11 precisely re-creates multi-page document structures and formatting, including text size and font styles, tables and diagrams, columns, headers, footers, footnotes, page numbers and more. A user-friendly interface

and pre-defined, automated tasks increase efficiency and eliminate the need for complicated routines and settings.

ABBYY Fine Reader based on three fundamental principles: Integrity, Purposefulness and Adaptability. The principle of integrity says that the observed object must always be considered as a "whole" consisting of many interrelated parts. The principle of purposefulness supposes that any interpretation of data must always serve some purpose. And the principle of adaptability means that the program must be capable of self-learning [104].

Let's take a look on how Fine Reader OCR recognizes text. First, the program analyzes the structure of document image. It divides the page into elements such as blocks of texts, tables, images, etc. The lines are divided into words and then into characters. Once the characters have been singled out, the program compares them with a set of pattern images. It advances numerous hypotheses about what this character is. Basing on these hypotheses the program analyzes different variants of breaking of lines into words and words into characters. After processing huge number of such probabilistic hypotheses, the program finally takes the decision, presenting you the recognized text [104].

ABBYY Fine Reader 11 detects any combination of 189 languages to help you expand your global capabilities. Fine Reader 11 supports a wide range of output formats. The results can also be sent directly to applications such as Microsoft Word, Excel and PowerPoint, Adobe Acrobat, Corel WordPerfect and OpenOffice.org Writer [105].

ABBYY Fine Reader is free for only 30 days and allows processing of up to 100 pages. It exports or saves maximum 3 pages to an external application at a time.

❖ **How it Works?**

I. Open- Open image or PDF file.

II. Read - Perform OCR.

III. Send - Save or Send document.

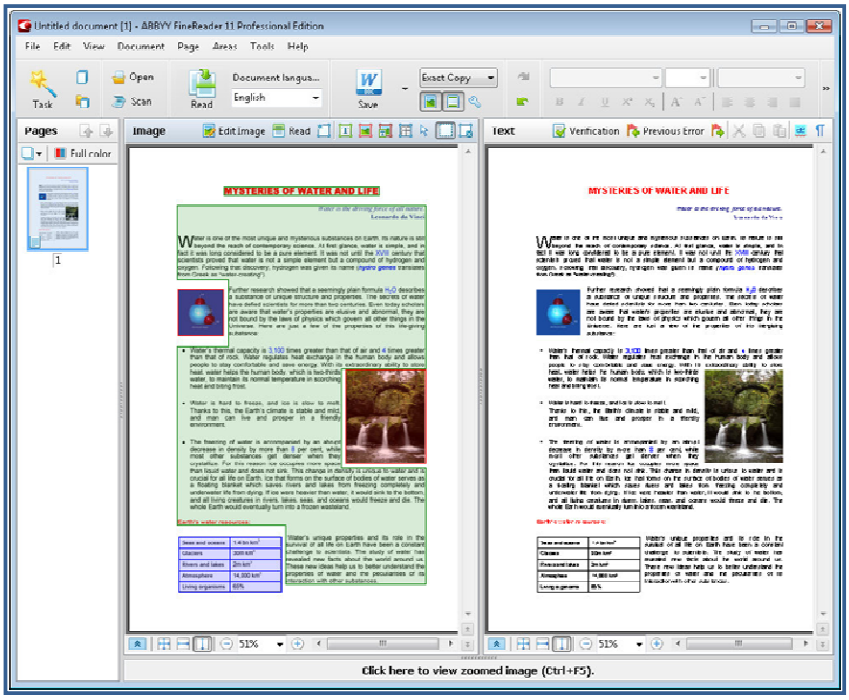

Figure 2.24 Screenshot of ABBYY Fine Reader 11

3. Transym OCR (TOCR)

Transym OCR is one of the proprietary optical character recognition tools, which provides the good amount of accuracy. Transym coverts the color images to gray scale images then does the OCR of these images. So we do not need to convert color image to grayscale while using Transym [106]. It reads broken, blurred and obscure characters. It allows to format output text.

❖ **How it Works?**

I. Open an image file from File menu.

II. OCR Image- To Perform OCR.

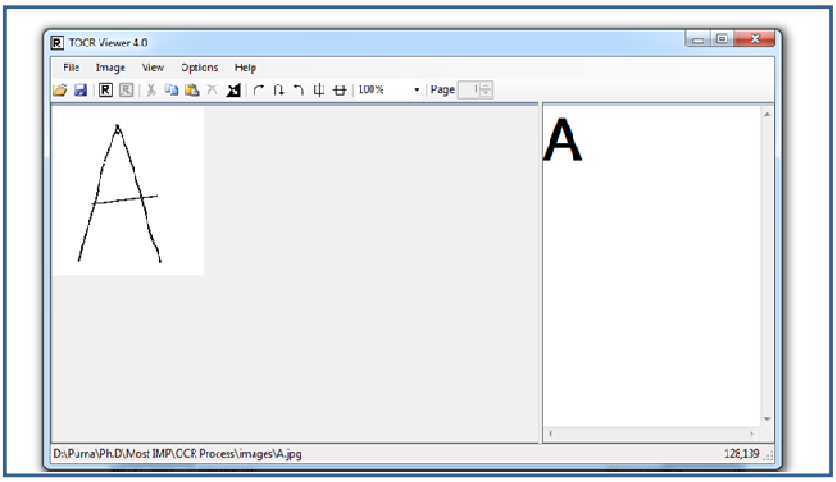

Figure 2.25 Screenshot of Transym OCR

4. Free OCR to Word

Free OCR to Word software is a simple and basic functionality OCR program. Free OCR to Word program works with any of the popular image files of JPG**,** JPEG**,** PSD**,** PNG**,** GIF**,** TIFF**,** BMP and more from a scanner attached to your computer or a digital camera and convert it into .DOC or .TXT format.

Free OCR to Word can recognize specified areas in an input image. You might need to select the image and text regions that are recognized by the OCR software. Some image & document tools are integrated with the software for image errors correcting and document editing. If the input image contains the orientation and skew errors, use RotateCW or RotateACW tools to fix the angle of them. For better viewing, try Fit Image**,** Fit Width as well as Zoom tools [107]. Free OCR to Word has a simple interface. So it is easy to use. It allows rotation of the image.

Free OCR to word cannot recognize formatting. It does not support PDF and multi-page file. It supports only one language.

❖ **How it Works?**

I. Open - Open an image file.

II. OCR – Convert an image file to text file.

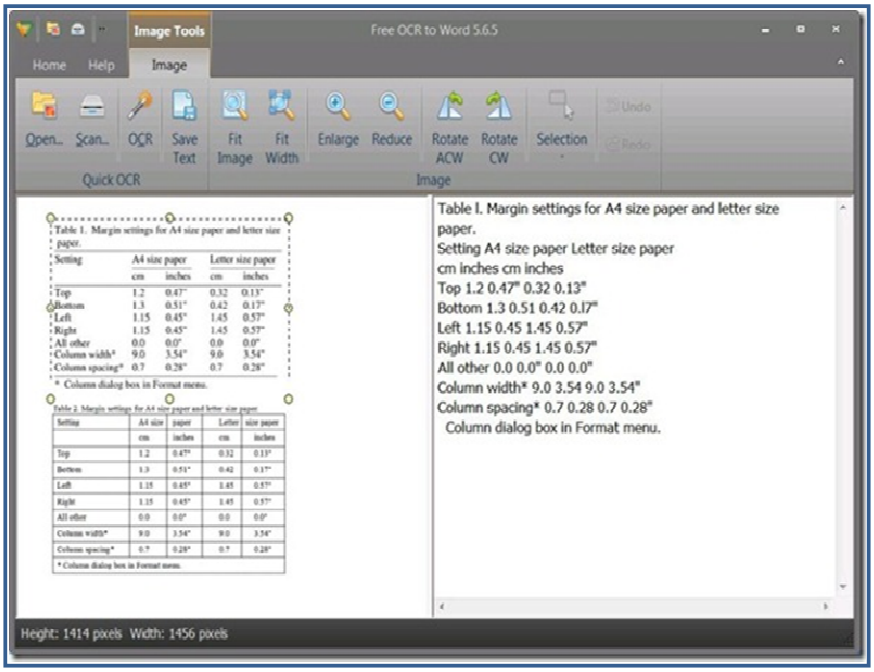

Figure 2.26 Screenshot of Free OCR to Word

5. Image to OCR converter

Image to OCR Converter is text recognition software that can read text from BMP, PDF, TIFF, JPG, GIF, PNG and all major image formats and saves the extracted text in WORD, DOC, PDF, HTML and TEXT formats with accurate text formatting and spacing. Complicated layout of legal documents, faxes, documents with tables, designs, and photos captured with digital and phone cameras are accurately recognized and recreated in output formats [108].

Image to OCR Converter recognizes more than 40 different languages. It retains logical structure and formatting elements in single and multi-page documents. Image to OCR Converter provides security features such as

password protection and watermark to the converted documents. The password protection prevents others from viewing or copying your document's content. Files can be watermarked to prevent illegal distribution. Image to OCR Converter provides automatic detection and correction of rotated, skewed and tilted documents. Broken text and characters are also reconstructed to provide better accuracy and recognition [108].

Image to OCR converter allows converting only two pages in unregistered version at a time. Security features such as password protection and watermark to the converted document are only provided in registration mode which is not free.

❖ **How it Works?**

I. Open File – Open an image file or PDF file.

II. Save File – Convert an image or PDF file to text file

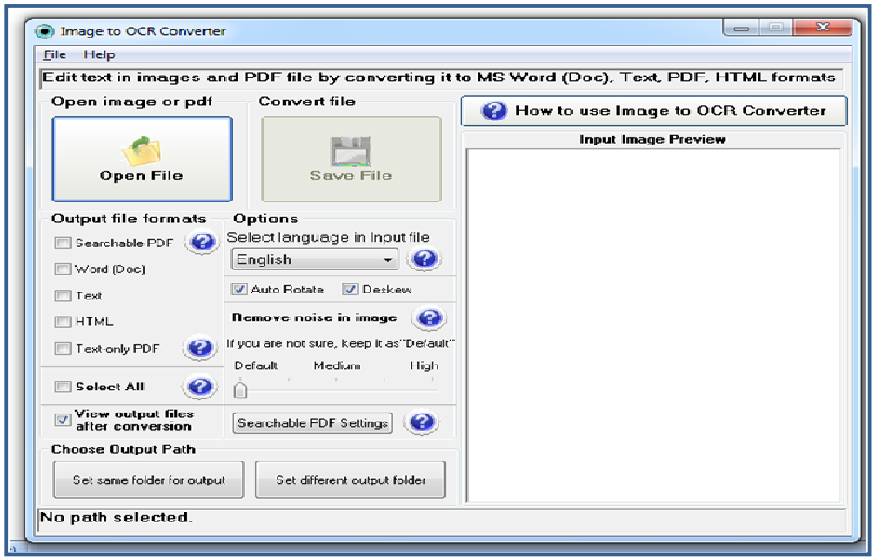

Figure 2.27 Screenshot of Image to OCR Conveter

6. A2ia

A2iA's OCR software recognizes isolated characters by distinguishing the individual shapes and sizes within each character classifier – identifying extracted such as curves, loops and lines- and organizing them in a logical and actionable manner according to the user's information management specification [109].

2.3.4 Other Web OCR Software

There are also other web OCR software available. Researcher has studied following web OCR software.

1. Google Docs
2. ABBYY Fine Reader
3. OCR Online
4. Online OCR

Above web OCR software are discussed in detail.

1. Google Docs

Google Docs is a free, web-based office suite offered by Google within its Google Drive service. It was formerly a storage service as well, but has since been replaced by Google Drive. It allows users to create and edit documents online while collaborating with other users live. Google Docs combines the features of Writely and Spreadsheets with a presentation program incorporating technology designed by Tonic Systems [110].

Data storage of files was introduced on January 12, 2010, with 1 GB of free space. On April 24, 2012, Google launched Google Drive which supplants Google Docs. Google Drive incorporates the Google Docs office suite into itself alongside providing improved storage functionality [110].

❖ **History**

Google Docs originated from two separate products, Writely and Google Spreadsheets. Writely was a web-based word processor created by the software company Upstartle and launched in August 2005.It was written by Sam Schillace and Steve Newman (both of whom had previously worked on Full Write and Claris Home Page) and Claudia Carpenter. They were trying out the then new Ajax technology and the "content editable" function in browsers, and intrigued by the idea of making a simpler version of Microsoft Word online [110].

Spreadsheets, launched as Google Labs Spreadsheets on June 6, 2006, originated from the acquisition of the XL2Web product by 2Web Technologies. Writely's original features included a collaborative text editing suite and access controls. Menus, keyboard shortcuts, and dialog boxes are similar to what users may expect in a desktop word processor such as Microsoft Word or LibreOffice Writer [110].

On March 9, 2006, Google announced that it had acquired Upstartle. At the time of acquisition, Upstartle had four employees. Writely closed registration to its service until the move to Google servers was complete. In August 2006, Writely sent account invitations to everyone who had requested to be placed on a waiting list, and then became publicly available on August 23. Writely continued to maintain its own user system until September 19, 2006, when it was integrated with Google Accounts [110].

Writely originally ran on Microsoft ASP.NET technology which uses Microsoft Windows. Since July 2006, Writely servers appear to be running a Linux-based operating system [110].

Meanwhile, Google developed Google Spreadsheets using the technology it had acquired from 2Web Technologies in 2005 and launched Google Labs

Spreadsheets on June 6, 2006, as the first public component of what would eventually become Google Docs. It was initially made available to only a limited number of users, on a first-come, first-served basis. The limited test was later replaced with a beta version available to all Google Account holders, around the same time as a press release was issued. In February 2007, Google Docs was made available to Google Apps users.

Google Docs is Google's "software as a service" office suite. Documents can be saved to a user's local computer in a variety of formats (ODF, HTML, PDF, RTF, Text, and Office Open XML). Documents are automatically saved to Google's servers, and a revision history is automatically kept so past edits may be viewed. Documents can be tagged and archived for organizational purposes. The service is officially supported on recent versions of the Firefox, Internet Explorer, Safari and Chrome browsers running on Microsoft Windows, Apple OS X and Linux operating systems [110].

Google Docs is one of many cloud computing document-sharing services. The majority of document-sharing services require user fees. (Google Docs is free for individuals, but has fees for business starting at $5/month.) [110].

Google introduced add-ons for Google Docs and Sheets which allow users to use third-party applications installed from the add-on stores to get additional features within the main services [110].

There are no limits on the number of files that you can process in a day. You can perform OCR on image and PDF files. It preserves most formatting. Documents, spreadsheets, presentations can be created with Google Docs, imported through the web interface, or sent via email.

With Google Docs, you can perform OCR on images and PDFs as large as 2 MB. For PDF files, they only look at the first 10 pages when searching for text to extract. Text in some languages may not be recognized and processed.

❖ **How it Works?**

I. Open Google Drive at drive.google.com

II. Select upload icon to upload any image or PDF file.

III. Click the setting icon in the top right and select upload settings.

IV. As shown in the Figure 2.28, select '*Convert documents, presentations, spreadsheets, and drawings to the corresponding Google Docs format'* and *'Convert text from PDF and image files to Google documents'* options.

V. Start upload - To perform OCR.

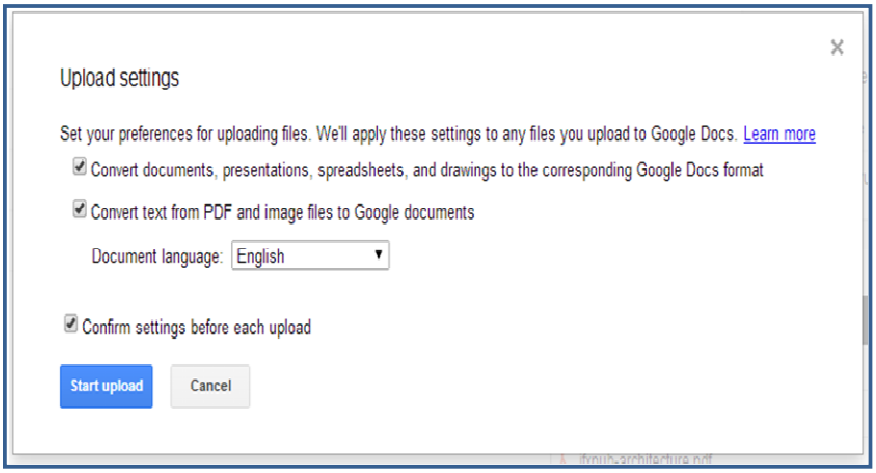

Figure 2.28 Upload Settings of Google Docs

If the OCR operation is successful, all the extracted text is stored as a new document else Google Docs will store your original image without any modification.

2. ABBYY Fine Reader

ABBYY Fine Reader also provides online service. Fine Reader read text from image and PDF format and saves the extracted text in .DOC, .DOCX, .XLS, .XLSX, .PDF, .RTF, .TXT and .ODT formats.

It supports maximum 30 MB file size. Fine Reader can convert multilingual document images to texts. You can choose up to 3 recognition languages for a multilingual document. Fine Reader can convert multi-page document and also preserves formatting. Fine Reader can export your converted file to Google Docs, Ever note and Drop Box.

❖ How it Works?

I. Upload – Upload an image or PDF file.

II. Select language and output format.

III. Recognize – To perform OCR.

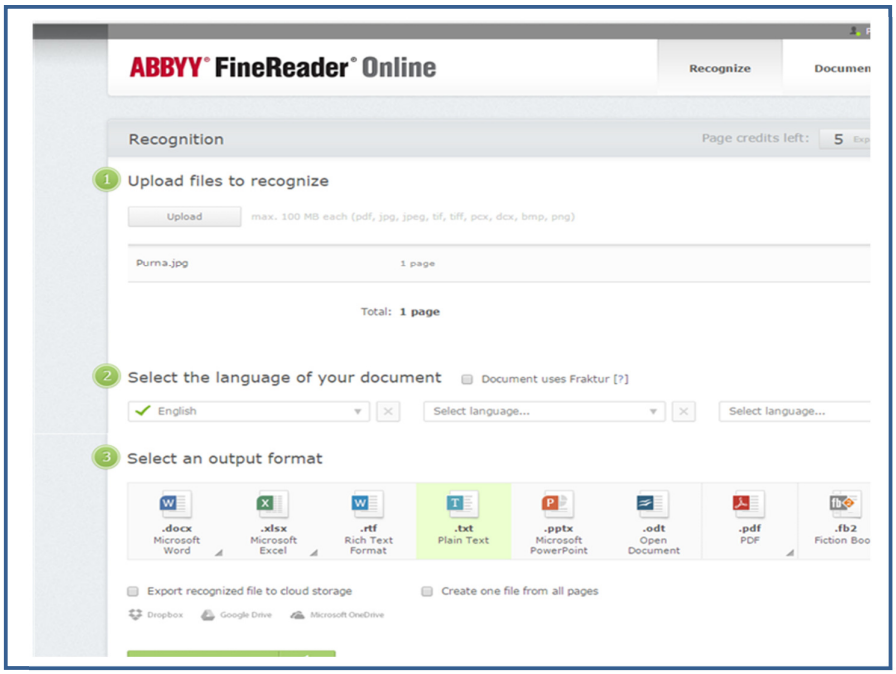

Figure 2.29 Screenshot of ABBYY Fine Reader Online

3. OCR Online

OCR Online is an online service for converting PDF and image file into editable and searchable text format. OCR Online read text from JPG, TIFF, PNG, GIF and PDF format and saves the extracted text in TXT, PDF, RTF and DOC format.

OCR Online has superior multilingual support and is capable of processing documents in 153 languages providing prime quality for a single language or any combination of the languages it supports [111]. It can rebuild the structure and restore formatting of multi-page documents. OCR Online gives the facility of batch processing. You can upload a large number of files in one batch and it will output the results as one document.

Only five pages are allowed to convert in a week. If you want to convert more than five pages a week, you have to pay for it. It supports maximum 10 MB file size.

❖ **How it Works?**

I. Browse - Select an image or PDF file.

II. Upload – upload an image or PDF file to perform OCR.

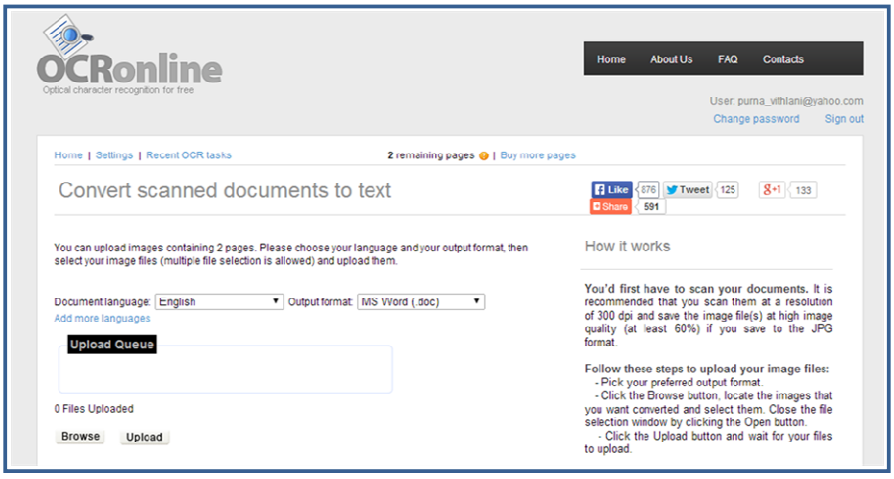

Figure 2.30 Screenshot of OCR Online

4. Online OCR

Online OCR is a free web-based Optical Character Recognition software (OCR) that allows you to convert scanned PDF documents (including multipage files), faxes, photographs or digital camera captured images into editable and searchable electronic documents including Adobe PDF, Microsoft Word, Microsoft Excel, Rtf, Html and Txt [112].

There are two modes in onlineOCR.net - Registered mode and Guest mode. Guest mode (without registration) allows you to convert 15 pages per hour with file size up to 4 MB.

Registration mode will give you access to additional features not available to guest users: recognition large images, ZIP archives and multipage PDF, choose recognition languages; convert into editable formats and other settings [112]. Registration mode allows you to convert 25 pages. For converting more than 25 pages, extra capacity may be purchased.

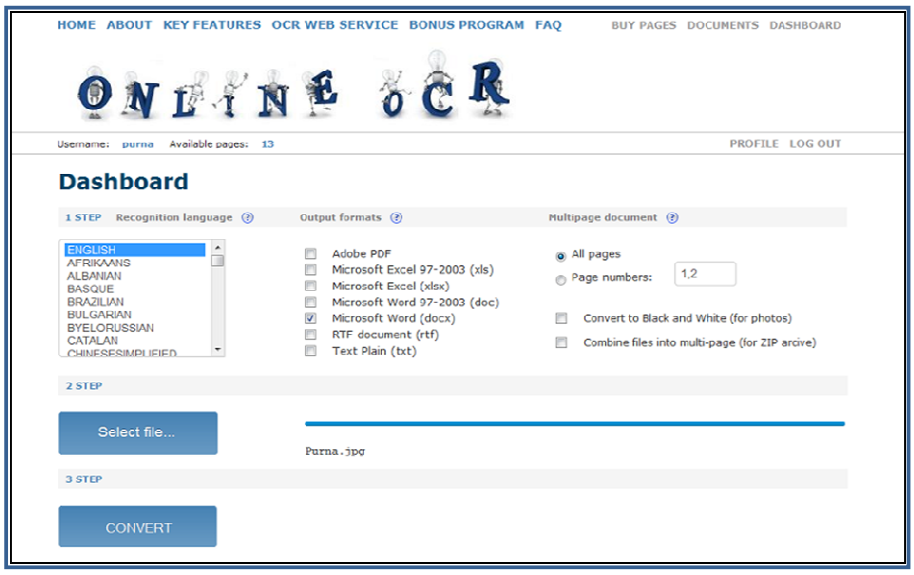

Figure 2.31 Screenshot of Online OCR

❖ **How it Works?**

I. Select File – Select an image file.

II. Output formats – Select output format.

III. Convert – Perform OCR.

2.4 Creation of Master Dataset for Characters and Digits

Researcher has created master dataset to analyze the performance of the above tools for handwritten character and digit recognition. Researcher has performed following steps.

2.4.1 Handwritten Data Sample Collection

Researcher has collected handwritten data samples for English capital characters A to Z and digits 0 to 9 from seven persons of different ages. Each character and digit is written 10 times by each person. Handwritten data samples are collected in A4 size blank paper. There are total 9 datasheets, 4 characters/digits are written 10 times on a single datasheet i.e. character A is written 5 times in 1st and 2nd row, character B in 3rd and 4th row, character C in 5th and 6th row and character D in 7th and 8th row so there are total 8 rows and 5 columns in each datasheet. Each datasheet contains 40 characters/digits.

Researcher has collected handwritten data samples from following persons.

Person Name - Purna, CKKSir, Mayur, Manoj, Manjulaben, Mitul, Jhanvi

Handwritten data samples are collected for capital characters in following manner.

A to Z - 70 samples of each character (Each character is written 10 time by 7 persons)

Handwritten data samples are collected for digit in following manner.

0 to 9 – 70 samples of each digit (Each digit is written 10 times by 7 persons)

2.4.2 Digitization of the Handwritten Datasheet

Character recognition tools require a scanned image as an input. To create an image file, handwritten datasheets are scanned using HP Deskjet 1510 scanner and saved in .jpg format. Figure 2.32 represents handwritten datasheet images of all capital characters and digits. Researcher has created total seven directories to store handwritten datasheet images of seven writers. Directories are named as per writer's name. i.e. Purna. Handwritten datasheet of English capital characters and digits are stored in the following format.

Writer's name_Characters in that datasheet

i.e. Purna_ABCD.jpg ……. Purna_7890.jpg

Table 2.1 Naming Convention of an Handwritten Datasheets

Naming Convention	Meaning
Purna_ABCD	File contains handwritten data samples of Character A, B, C and D written 10 times by writer Purna.
Purna_EFGH	File contains handwritten data samples of Character E, F, G and H written 10 times by writer Purna.
Purna_IJKL	File contains handwritten data samples of Character I, J, K and L written 10 times by writer Purna.
Purna_MNOP	File contains handwritten data samples of Character M, N, O and P written 10 times by writer Purna.
Purna_QRST	File contains handwritten data samples of Character Q, R, S and T written 10 times by writer Purna.
Purna_UVWX	File contains handwritten data samples of Character U, V, W and X written 10 times by writer Purna.
Purna_YZ12	File contains handwritten data samples of Character Y and Z and digit 1 and 2 written 10 times by writer Purna.
Purna_3456	File contains handwritten data samples of digit 3, 4, 5 and 6 written 10 times by writer Purna.
Purna_7890	File contains handwritten data samples of digit 7, 8, 9 and 0 written 10 times by writer Purna.

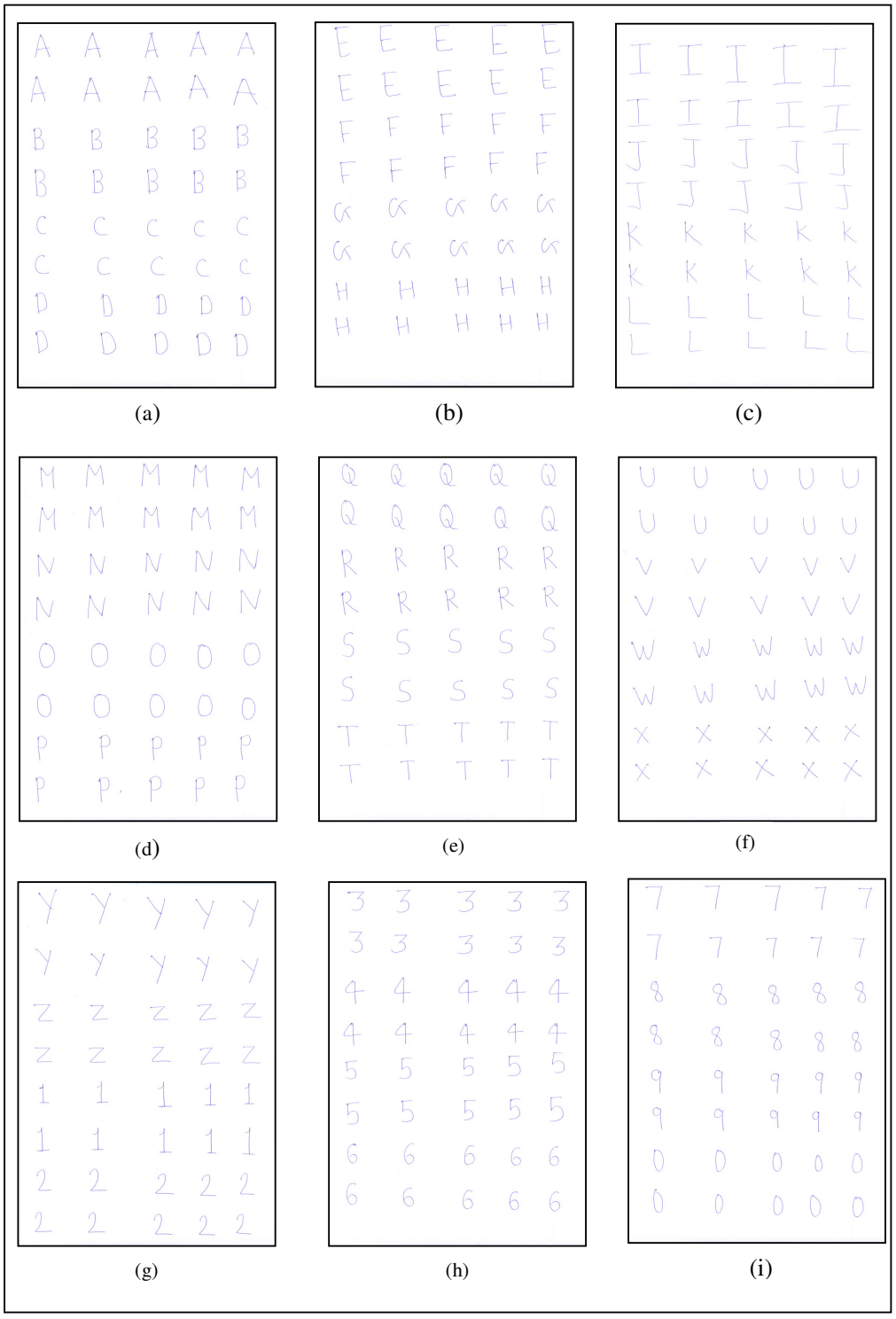

Figure 2.32 Sample Handwritten Datasheets for Experimental Study (a) A B C D (b) E F G H (c) I J K L (d) M N O P (e) Q R S T(f) U V W X (g) Y Z 1 2 (h) 3 4 5 6 (i) 7 8 9 0

2.5 Result Analysis of Tools for Characters and Digits

Among above tools, Researcher has selected six tools named Simple OCR, Free OCR to Word, Image to OCR Converter, Free OCR, Custom OCR Online and PDF OCR X. Result analysis for characters and digits for these six tools is presented in Table 2.2 to Table 2.13. Columns represent Total recognized characters out of 70 handwritten character samples, Recognition rate and Recognition rate is less than 35% or not. Recognition rate of each character and digit is calculated using following equation.

Total Number of recognized Character/Digit × 100 / 70

1. Simple OCR – Characters

Result analysis of characters in simple OCR is presented in Table 2.2.

Table 2.2 Result Analysis of Characters in Simple OCR

Characters	Total Recognize Characters out of 70	Recognition Rate (%)	<35
A	6	8.57	Y
B	11	15.71	Y
C	20	28.57	Y
D	9	12.86	Y
E	4	5.71	Y
F	21	30	Y
G	5	7.14	Y
H	1	1.43	Y
I	48	68.57	N
J	7	10	Y
K	25	35.71	N
L	5	7.14	Y
M	25	35.71	N
N	0	0	Y

O	12	17.14	Y
P	41	58.57	N
Q	4	5.71	Y
R	7	10	Y
S	35	50	N
T	8	11.43	Y
U	35	50	N
V	35	50	N
W	28	40	N
X	36	51.43	N
Y	20	28.57	Y
Z	24	34.29	Y

2. Simple OCR – Digits

Result analysis of digits in simple OCR is presented in Table 2.3.

Table 2.3 Result Analysis of Digits in Simple OCR

Digits	Total Recognized Digits out of 70	Recognition Rate (%)	<35
1	37.14	37.14	N
2	47.14	47.14	N
3	68.57	68.57	N
4	17.14	17.14	Y
5	64.29	64.29	N
6	32.86	32.86	Y
7	71.43	71.43	N
8	51.43	51.43	N
9	68.57	68.57	N
0	28.57	28.57	Y

3. Free OCR to Word – Characters

Result analysis of characters in Free OCR to Word is presented in Table 2.4.

Table 2.4 Result Analysis of Characters in Free OCR to Word

Characters	Recognized Characters out of 70	Recognition Rate (%)	<35
A	24	34.29	Y
B	11	15.71	Y
C	23	32.86	Y
D	35	50	N
E	11	15.71	Y
F	10	14.29	Y
G	1	1.43	Y
H	21	30	Y
I	15	21.43	Y
J	17	24.29	Y
K	18	25.71	Y
L	36	51.43	N
M	18	25.71	Y
N	13	18.57	Y
O	3	4.29	Y
P	24	34.29	Y
Q	44	62.86	N
R	15	21.43	Y
S	6	8.57	Y
T	16	22.86	Y
U	44	62.86	N
V	34	48.57	N
W	29	41.43	N
X	17	24.29	Y
Y	13	18.57	Y
Z	25	35.71	N

4. Free OCR to Word – Digits

Result analysis of digits in Free OCR to Word is presented in Table 2.5.

Table 2.5 Result analysis of Digits in Free OCR to Word

Digits	Recognized Digits out of 70	Recognition Rate (%)	<35
1	18	25.71	Y
2	10	14.29	Y
3	14	20	Y
4	5	7.14	Y
5	15	21.43	Y
6	4	5.71	Y
7	12	17.14	Y
8	4	5.71	Y
9	7	10	Y
0	9	12.86	Y

5. Image to OCR Converter – Characters

Result analysis of characters in Image to OCR Converter is presented in Table 2.6.

Table 2.6 Result Analysis of Characters in Image to OCR Converter

Characters	Recognized Characters out of 70	Recognition Rate (%)	<35
A	39	55.71	N
B	27	38.57	N
C	38	54.29	N
D	34	48.57	N
E	38	54.29	N
F	44	62.86	N
G	5	7.14	Y

H	25	35.71	N
I	21	30	Y
J	44	62.86	N
K	31	44.29	N
L	50	71.43	N
M	32	45.71	N
N	15	21.43	Y
O	8	11.43	Y
P	40	57.14	N
Q	32	45.71	N
R	30	42.86	N
S	29	41.43	N
T	39	55.71	N
U	57	81.43	N
V	62	88.57	N
W	31	44.29	N
X	23	32.86	Y
Y	3	4.29	Y
Z	21	30	Y

6. Image to OCR Converter - Digits

Result analysis of digits in Image to OCR Converter is presented in Table 2.7.

Table 2.7 Result Analysis of Digits in Image to OCR Converter

Digits	Recognized Digits out of 70	Recognition Rate (%)	<35
1	16	22.86	Y
2	12	17.14	Y
3	41	58.57	N
4	16	22.86	Y

5	34	48.57	N
6	19	27.14	Y
7	16	22.86	Y
8	1	1.43	Y
9	12	17.14	Y
0	46	65.71	N

7. Free OCR - Characters

Result analysis of characters in Free OCR is presented in Table 2.8.

Table 2.8 Result Analysis of Characters in Free OCR

Characters	Recognized Characters out of 70	Recognition Rate (%)	<35
A	41	58.57	N
B	24	34.29	Y
C	37	52.86	N
D	45	64.29	N
E	35	50	N
F	31	44.29	N
G	5	7.14	Y
H	41	58.57	N
I	32	45.71	N
J	17	24.29	Y
K	46	65.71	N
L	49	70	N
M	35	50	N
N	23	32.86	Y
O	9	12.86	Y
P	30	42.86	N
Q	38	54.29	N

R	25	35.71	N
S	15	21.43	Y
T	33	47.14	N
U	38	54.29	N
V	18	25.71	Y
W	31	44.29	N
X	28	40	N
Y	10	14.29	Y
Z	41	58.57	N

8. Free OCR - Digits

Result analysis of digits in Free OCR is presented in Table 2.9.

Table 2.9 Result Analysis of Digits in Free OCR

Digits	Recognized Digits out of 70	Recognition Rate (%)	<35
1	20	28.57	Y
2	9	12.86	Y
3	21	30	Y
4	8	11.43	Y
5	23	32.86	Y
6	19	27.14	Y
7	16	22.86	Y
8	1	1.43	Y
9	4	5.71	Y
0	9	12.86	Y

9. Custom OCR Online - Characters

Result analysis of Characters in Custom OCR Online is presented in Table 2.10.

Table 2.10 Result Analysis of Characters in Custom OCR Online

Characters	Recognized Characters out of 70	Recognition Rate (%)	<35
A	45	64.29	N
B	31	44.29	N
C	25	35.71	N
D	44	62.86	N
E	47	67.14	N
F	36	51.43	N
G	6	8.57	Y
H	45	64.29	N
I	34	48.57	N
J	16	22.86	Y
K	38	54.29	N
L	50	71.43	N
M	46	65.71	N
N	15	21.43	Y
O	12	17.14	Y
P	55	78.57	N
Q	45	64.29	N
R	31	44.29	N
S	11	15.71	Y
T	46	65.71	N
U	53	75.71	N
V	25	35.71	N
W	44	62.86	N
X	17	24.29	Y
Y	14	20	Y
Z	52	74.29	N

10. Custom OCR Online - Digits

Result analysis of digits in Custom OCR Online is presented in Table 2.11.

Table 2.11 Result Analysis of Digits in Custom OCR Online

Digits	Recognized Digits out of 70	Recognition Rate (%)	<35
1	31	44.29	N
2	10	14.29	Y
3	26	37.14	N
4	22	31.43	Y
5	46	65.71	N
6	26	37.14	N
7	39	55.71	N
8	4	5.71	Y
9	6	8.57	Y
0	13	18.57	Y

11. PDF OCR X - Characters

Result analysis of Characters in PDF OCR X is presented in Table 2.12.

Table 2.12 Result analysis of Characters in PDF OCR X

Characters	Recognized Characters out of 70	Recognition Rate (%)	<35
A	2	2.86	Y
B	0	0	Y
C	4	5.71	Y
D	1	1.43	Y
E	9	12.86	Y
F	4	5.71	Y
G	0	0	Y
H	6	8.57	Y

I	0	0	Y
J	0	0	Y
K	0	0	Y
L	0	0	Y
M	2	2.86	Y
N	2	2.86	Y
O	0	0	Y
P	1	1.43	Y
Q	11	15.71	Y
R	7	10	Y
S	5	7.14	Y
T	8	11.43	Y
U	23	32.86	Y
V	22	31.43	Y
W	23	32.86	Y
X	13	18.57	Y
Y	0	0	Y
Z	0	0	Y

12. PDF OCR X - Digits

Result analysis of digits in PDF OCR X is presented in Table 2.13.

Table 2.13 Result Analysis of Digits in PDF OCR X

Digits	Recognized Digits out of 70	Recognition Rate (%)	<35
1	0	0	Y
2	0	0	Y
3	2	2.86	Y
4	5	7.14	Y
5	3	4.29	Y

6	2	2.86	Y
7	4	5.71	Y
8	0	0	Y
9	0	0	Y
0	0	0	Y

2.6 Summary of Characters and Digits for Evaluated Tools

Researcher has analyzed Table 2.2 to 2.13 and found out the highest recognition rate of characters/digits among those tools. Recognition rate and highest recognition rate of character/digit among those tools is presented in Table 2.14.

Table 2.14 Recognition Rate (In Percentage) of each Character and Digit using Character Recognition Tools

Character /Digit	Character Recognition Tools						Highest Recognition Rate
	Simple OCR	Free OCR to Word	Image to OCR Converter	Free OCR	Custom OCR Online	PDF OCR X	
A	8.57	34.29	55.71	58.57	**64.29**	2.86	**64.29**
B	15.71	15.71	38.57	34.29	**44.29**	0	**44.29**
C	28.57	32.86	**54.29**	52.86	35.71	5.71	**54.29**
D	12.86	50	48.57	**64.29**	62.86	1.43	**64.29**
E	5.71	15.71	54.29	50	**67.14**	12.86	**67.14**
F	30	14.29	**62.86**	44.29	51.43	5.71	**62.86**
G	7.14	1.43	7.14	7.14	**8.57**	0	**8.57**
H	1.43	30	35.71	58.57	**64.29**	8.57	**64.29**
I	**68.57**	21.43	30	45.71	48.57	0	**68.57**
J	10	24.29	**62.86**	24.29	22.86	0	**62.86**
K	35.71	25.71	44.29	**65.71**	54.29	0	**65.71**
L	7.14	51.43	**71.43**	70	**71.43**	0	**71.43**
M	35.71	25.71	45.71	50	**65.71**	2.86	**65.71**
N	0	18.57	21.43	**32.86**	21.43	2.86	**32.86**
O	**17.14**	4.29	11.43	12.86	**17.14**	0	**17.14**

P	58.57	34.29	57.14	42.86	**78.57**	1.43	**78.57**
Q	5.71	62.86	45.71	54.29	**64.29**	15.71	**64.29**
R	10	21.43	42.86	35.71	**44.29**	10	**44.29**
S	**50**	8.57	41.43	21.43	15.71	7.14	**50**
T	11.43	22.86	55.71	47.14	**65.71**	11.43	**65.71**
U	50	62.86	**81.43**	54.29	75.71	32.86	**81.43**
V	50	48.57	**88.57**	25.71	35.71	31.43	**88.57**
W	40	41.43	44.29	44.29	**62.86**	32.86	**62.86**
X	**51.43**	24.29	32.86	40	24.29	18.57	**51.43**
Y	**28.57**	18.57	4.29	14.29	20	0	**28.57**
Z	34.29	35.71	30	58.57	**74.29**	0	**74.29**
1	37.14	25.71	22.86	28.57	**44.29**	0	**44.29**
2	**47.14**	14.29	17.14	12.86	14.29	0	**47.14**
3	**68.57**	20	58.57	30	37.14	2.86	**68.57**
4	17.14	7.14	22.86	11.43	**31.43**	7.14	**31.43**
5	64.29	21.43	48.57	32.86	**65.71**	4.29	**65.71**
6	32.86	5.71	27.14	27.14	**37.14**	2.86	**37.14**
7	**71.43**	17.14	22.86	22.86	55.71	5.71	**71.43**
8	**51.43**	5.71	1.43	1.43	5.71	0	**51.43**
9	**68.57**	10	17.14	5.71	8.57	0	**68.57**
0	28.57	12.86	**65.71**	12.86	18.57	0	**65.71**

Custom OCR Online gives the best result because it identifies 14 characters and 4 digits which is the highest recognition rate among the six evaluated tools. As shown in Highest Recognition Rate column of above Table, recognition rate of handwritten characters and digits in evaluated six tools is shown in Table 2.15 and Table 2.16.

Table 2.15 Highest Recognition Rate of Characters

Character Recognition Tool	Characters	Total Characters
Simple OCR	I,S,X,Y,	4
Free OCR to Word	-	0
Image to OCR Converter	C,F,J,L,U,V	6
Free OCR	D,K,N	3
Custom OCR Online	A,B,E,G,H,L,M,O,P,Q,R,T,W,Z	14
PDF OCR X	-	0

Figure 2.33 shows Column chart representation of above Table. As shown in the chart, the Custom OCR Online gives the highest recognition rate for 14 characters.

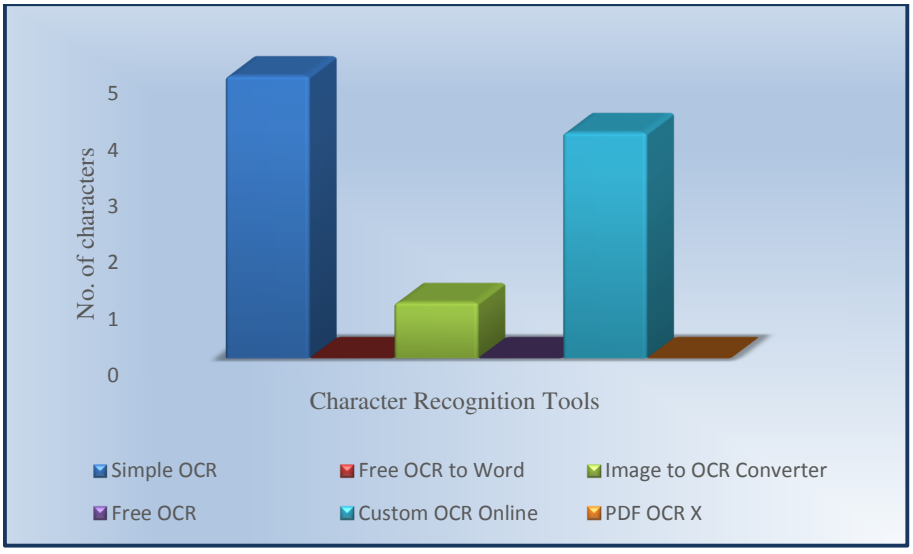

Figure 2.33 Column Chart Representation of Highest Recognition Rate of Characters

Table 2.16 Highest Recognition Rate of Digits

Character Recognition Tool	Digits	Total Digits
Simple OCR	2,3,7,8,9	5
Free OCR to Word	-	0
Image to OCR Converter	0	1
Free OCR	-	0
Custom OCR Online	1,4,5,6	4
PDF OCR X	-	0

Figure2.34shows Column chart representation of above Table. As shown in the chart, the Simple OCR gives the highest recognition rate for 5 digits.

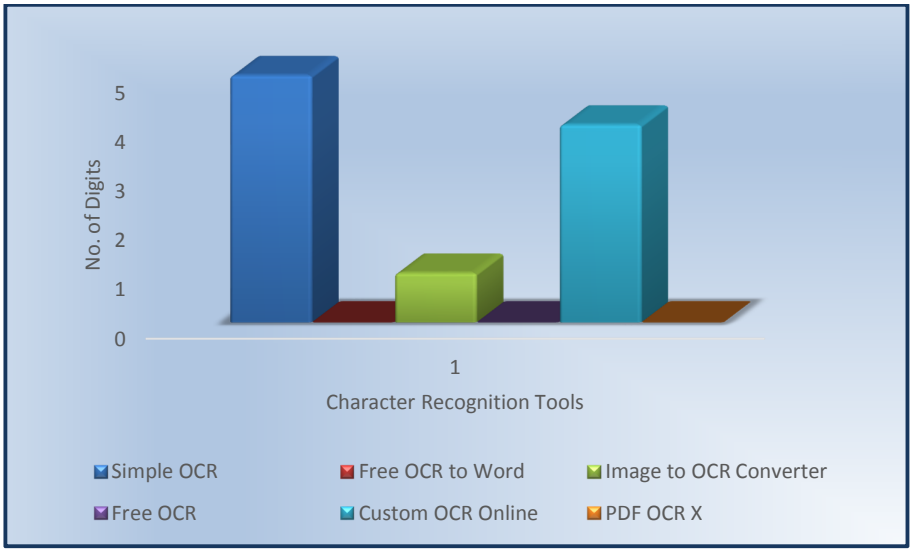

Figure 2.34 Column Chart Representation of Highest Recognition Rate of Digits

2.7 Recognition Rate Analysis

To find the deficiency in identifying the patterns of optical character, Researcher has created dataset of handwritten characters and digits as described in Section 2.4, counted the number of recognized characters/digits,

calculated the recognition rate and found out the highest recognition rate of character/digit among evaluated tools. On the basis of above work, researcher has found out the characters and digits whose recognition rate is less than 35% in all evaluated tools which is shown in Table 2.17 and 2.18.

Table 2.17 Characters below 35% Recognition Rate

Character	Character Recognition Tools					
	Simple OCR	Free OCR to Word	Image to OCR Converter	Free OCR	Custom OCR Online	PDF OCR X
G	7.14	1.43	7.14	7.14	8.57	0
N	0	18.57	21.43	32.86	21.43	2.86
O	17.14	4.29	11.43	12.86	17.14	0
Y	28.57	18.57	4.29	14.29	20	0

Table 2.18 Digits below 35% Recognition Rate

Digit	Character Recognition Tools					
	Simple OCR	Free OCR to Word	Image to OCR Converter	Free OCR	Custom OCR Online	PDF OCR X
4	17.14	7.14	22.86	11.43	31.43	7.14

As shown in Table 2.17 and 2.18, recognition rate of characters **G, N, O, Y** and digit **4** is less than 35% in all evaluated tools.

❖ **References**

[1] http://en.wikipedia.org/wiki/Handwriting_recognition

[2] Suruchi G. Dedgaonkar, Anjali A. Chandavale, Ashok M. Sapkal, "Survey of Methods for Character recognition", International

Journal of Engineering and Innovative Technology (IJEIT),Volume 1, Issue 5, May 2012.

[3] Nadira Muda, Nik Kamariah Nik Ismail, Siti Azami Abu Bakar, Jasni Mohamad Zain Fakulti Sistem Komputer & Kejuruteraan Perisian, **"**Optical Character Recognition By Using Template Matching", University Malaysia Pahang Karung Berkunci 12, 25000 Kuantan, Pahang.

[4] www.academia.edu/714194/Optical_Character_Recognition_By_Using_Template_Matching_Alphabet_

[5] Faisal Mohammad, Jyoti Anarase, Milan Shingote and Pratik Ghanwat, "Optical Character Recognition Implementation Using Pattern Matching", International Journal of Computer Science and Information Technologies, Vol. 5(2), 2014.

[6] Mo Wenying and Ding Zuchun, "A Digital Character Recognition Algorithm Based on the Template Weighted Match Degree", Vol. 18, pp. 53-60, 2013.

[7] Mr. Danish Nadeem and Miss. Saleha Rizvi, "Character Recognition using Template Matching", Department of computer Science, JMI.

[8] Rachit Virendra Adhvaryu, "Optical Character Recognition using Template Matching", International Journal of Computer Science Engineering and Information Technology Research, Vol. 3, Issue 4, Oct 2013.

[9] M. Ziaratban, K. Faez, F. Faradji, "Language-based feature extraction using template-matching in Farsi/Arabic handwritten numeral recognition", Ninth International Conference on Document Analysis and Recognition, pp. 297 - 301, 2007.

[10] Sumedha B. Hallale, Geeta D. Salunke, "Twelve Directional Feature Extraction for Handwritten English Character Recognition", International Journal of Recent Technology and Engineering, Vol. 2, Issue-2, May 2013.

[11] Rohit Verma and Dr. Jahid Ali, "A-Survey of Feature Extraction and Classification Techniques in OCR Systems", International Journal of

Computer Applications & Information Technology, Vol. 1, Issue 3, November 2012.

[12] http://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm

[13] Yasmine Elglaly, Francis Quek, “Isolated Handwritten Arabic Characters Recognition using Multilayer Perceptrons and K Nearest Neighbour Classifiers”.

[14] Rumiana Krasteva, “Bulgarian Hand-Printed Character Recognition Using Fuzzy C-Means Clustering”, Problems of engineering and robotics”, pp. 112-117.

[15] http://link.springer.com/chapter/10.1007%2F978-3-642-05253-8_33#page-1

[16] M. S. Ryan and G. R. Nudd., “Dynamic Character Recognition Using Hidden Markov Models”, May 1993.

[17] Ganesh S Pawar and Sunil S Morade, “Realization of Hidden Markov Model for English Digit Recognition”, IJCA, Vol. 98– No.17, July 2014.

[18] Rajib Lochan Das, Binod Kumar Prasad, Goutam Sanyal, "HMM based Offline Handwritten Writer Independent English Character Recognition using Global and Local Feature Extraction", International Journal of Computer Applications (0975 8887), Volume 46 No.10, pp. 45-50, May 2012.

[19] Pritpal Singh and Sumit Budhiraja, “Feature Extraction and Classification Techniques in O.C.R. Systems for Handwritten Gurmukhi Script”. International Journal of Engineering Research and Applications (IJERA), Vol.1, ISSUE 4, pp.1736-1739.

[20] N. Suguna and Dr. K. Thanushkodi, “An Improved k-Nearest Neighbour Classification Using Genetic Algorithm”, IJCSI, Vol. 7, Issue 4, No 2, July 2010.

[21] Emanuel Indermuhle, Marcus Liwicki and Horst Bunke, “Recognition of Handwritten Historical Documents: HMM-Adaptation vs. Writer Specific Training”.

[22] B.V.Dhandra , Gururaj Mukarambi and Mallikarjun Hangarge, "Handwritten Kannada Vowels and English Character Recognition System", International Journal of Image Processing and Vision Sciences,Vol. 1 Issue1 ,2012.

[23] S. Antani, L. Agnihotri, Guajarati character recognition, Proceedings of the Fifth International Conference on Document Analysis and Recognition, 1999, pp. 418–421.

[24] Avani R. Vasant, Sandeep R. Vasant and Dr. G.R. Kulkarni, "Gujarati Character Recognition: The State of the art Comprehensive Survey", Journal of Information, Knowledge and Research in Computer Engineering, Vol. 02, Issue 01.

[25] Noha A. Yousri, Mohamed S. Kamel and Mohamed A. Ismail, "Finding Arbitrary Shaped Clusters for Character Recognition".

[26] Richa Goswami and O.P. Sharma, "A Review on Character Recognition Techniques", IJCA, Vol. 83, No. 7, December 2013.

[27] Ms. M. Shalini, Dr. B. Indira, "Automatic Character Recognition of Indian Languages – A brief Survey", IJISET, Vol. 1, Issue 2, April 2014.

[28] Jonathan J. HULL, Alan COMMIKE and Tin-Kam HO, "Multiple Algorithms for Handwritten Character Recognition".

[29] Hewavitharana, S, and H.C. Fernando, 2002. "A Two-Stage Classification Approach to Tamil Handwriting Recognition", pp: 118-124, Tamil Internet 2002, California, USA.

[30] R. Jagadeesh Kannan and R. Prabhakar, "A Comparative Study of Optical Character Recognition for Tamil Script", European Journal of Scientific Research, Vol.35 No.4 (2009), pp.570-582.

[31] José C. Principe, Neil R. Euliano, Curt W. Lefebvre "Neural and Adaptive Systems: Fundamentals Through Simulations", ISBN 0-471-35167-9

[32] http://www.heatonresearch.com/online/introduction-neural-networks-cs-edition-2/chapter-5/page4.html

[33] http://shodhganga.inflibnet.ac.in/bitstream/10603/48/6/chaper%204_c%20b%20bangal.pdf

[34] http://www.heatonresearch.com/online/introduction-neural-networks-cs-edition-2/chapter-5/page1.html

[35] http://en.wikipedia.org/wiki/Recurrent_neural_network

[36] http://en.wikipedia.org/wiki/Self-organizing_map

[37] Fiona Nielsen, "Neural Networks – algorithms and applications", 2001.

[38] Dayashankar Singh, J.P. Saini and D.S. Chauhan, "Analysis of Handwritten Hindi Character Recognition using Advanced Feature Extraction Technique and Back propagation Neural Network", International Journal of Computer Applications, Vol. 97, No.22, July 2014.

[39] Neural Computing Theory and Practices by Philip D.Wasserman

[40] Neural Networks, Fuzzy Logic, and Genetic Algorithms by S. Rajasekaran and G.A. VijaylakshmiPai.

[41] http://en.wikipedia.org/wiki/Multilayer_perceptron.

[42] Rosenblatt, Frank. x. Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms. Spartan Books, Washington DC, 1961.

[43] Rumelhart, David E., Geoffrey E. Hinton, and R. J. Williams. "Learning Internal Representations by Error Propagation". David E. Rumelhart, James L. McClelland, and the PDP research group. (Editors), Parallel distributed processing: Explorations in the microstructure of cognition, Volume 1: Foundations. MIT Press, 1986.

[44] Cybenko, G. 1989. Approximation by super positions of a sigmoidal function Mathematics of Control, Signals, and Systems, 2(4), 303–314.

[45] http://wwwold.ece.utep.edu/research/webfuzzy/docs/kk-thesis/kk-thesis-html/node22.html.

[46] http://en.wikipedia.org/wiki/Backpropagation.

[47] Parveen Kumar, Nitin Sharma and Arun Rana, “Handwritten Character Recognition using Different Kernel based SVM Classifier and MLP Neural Network”, International Journal of Computer Science, Vol. 53, No.11, Sep.2012.

[48] Yusuf Perwej and Ashish Chaturvedi, “Neural Networks for Handwritten English Alphabet Recognition”, International Journal of Computer Application, Vol. 2, No. 7, April 2011.

[49] Sameeksha Barve, “Optical Character Recognition using Artificial Neural Network”, International Journal of Advanced Technology and Engineering Research, Vol. 2, Issue 2, May 2012.

[50] MdFazlul Kader and Kaushik Deb, “Neural Network based English alphanumeric character recognition, International Journal of Computer Science, Engineering and Applications, Vol.2, No.4, August 2012.

[51] J. Pradeep, E. Srinivasan, S. Himavathib, "Neural Network Based Recognition System Integrating Feature Extraction and Classification for English Handwritten", International journal of Engineering,Vol.25, No. 2, pp. 99-106, May 2012.

[52] Jitendra shrivastav and Ravindra Gupta, “A Survey of Modern Character Recognition Techniques”, International Journal of Scientific Research and Development, Vol.1, Issue 2, 2013.

[53] Velappa Ganapathy, Kok Leong Liew, "Handwritten Character Recognition Using Multi scale Neural Network Training Technique", World Academy of Science, Engineering and Technology, pp. 32-37, 2008.

[54] Md. Alamgir Badsha, Md. Akash Ali, Dr. Kaushik Deb and Md. Nuruzzaman Bhuiyan, “Handwritten Bangla Character Recognition Using Neural Network”, International Journal of Advanced Research in Computer Science and Software Engineering, Vol. 2, Issue 11, Nov.2012.

[55] V. Kalaichelvi and Ahammed Shamir, "Application of Neural Networks in Character Recognition", International Journal of Computer Application, Vol. 52, No.12, August 2012.

[56] Christopher M. Bishop, "Pattern Recognition and Machine Learning", Springer Publication, Singapore, 2006, Pp. 1-3, 308-320.

[57] Anshuman Sharma, "Handwritten digit Recognition using Support Vector Machine".

[58] http://cdn.intechopen.com/pdfs-wm/40722.pdf

[59] Richa Goswami and O.P. Sharma, "A Review on Character Recognition Techniques", IJCA, Vol. 83, No. 7, December 2013.

[60] Sandhya Arora, Debotosh Bhattacharjee, Mita Nasipuri, L. Malik, M. Kundu and D. K. Basu, "Performance Comparison of SVM and ANN for Handwritten Devnagari Character Recognition", International Journal of Computer Science Issues, Vol. 7, Issue 3, May 2010

[61] C. J. C. Burges, .A tutorial on support vector machines for pattern recognition, Data Mining and Knowledge Discovery, 1998, pp. 121-167.

[62] Antonio Carlos Gay Thomé, "SVM Classifiers – Concepts and Applications to Character Recognition".

[63] Jonathan Milgram, Mohamed Cheriet and Robert Sabourin, "One against One" or "One against All": Which One is better for Handwriting Recognition with SVMs?"

[64] Dewi Nasien, Habibollah Haron & Siti Sophiayati Yuhaniz, Support Vector Machine (SVM) for English Handwritten Character Recognition, Second International Conference on Computer Engineering and Applications, 2010.

[65] Indrani Bhattacherjee, "Off-line English Character Recognition: A Comparative Survey", Proc. of Int. Conf. on Recent Trends in Information, Telecommunication and Computing 2013.

[66] Shailedra Kumar Shrivastava and Sanjay S. Gharde, "Support Vector Machine for Handwritten Devanagari Numeral Recognition",

International Journal of Computer Application, Vol.7, No.11, Oct. 2010.

[67] Munish Kumar, M. K. Jindal, R. K. Sharma, "SVM Based Offline Handwritten Gurmukhi Character Recognition".

[68] Parveen Kumar, Nitin Sharma and Arun Rana, "Handwritten Character Recognition using Different Kernel based SVM Classifier and MLP Neural Network", International Journal of Computer Science, Vol. 53, No.11, Sep.2012.

[69] ZHAO Bin, LIU Yong and XIA Shao Wei "Support Vector Machine and its Application in Handwritten Numeral Recognition" 0-7695-0750-6/00 2000 IEEE.

[70] Gita Sinha, Anita Rani, Prof. Renu Dhir, Mrs. Rajneesh Rani "Zone-Based Feature Extraction Techniques and SVM for Handwritten Gurmukhi Character Recognition", IJARCSSE, Vol. 2, Issue 6, June 2012.

[71] N. Shanthi and K. Duraiswamy, "Performance Comparison of Different Image Sizes for Recognizing Unconstrained Handwritten Tamil Characters using SVM".

[72] R. Jagadeesh Kannan and R. Prabhakar, "A Comparative Study of Optical Character Recognition for Tamil Script", European Journal of Scientific Research, Vol.35 No.4 (2009), pp.570-582.

[73] Support Vector Machine Based Gujarati Numeral Recognition Mamta Maloo, K. V. Kale International Journal on Computer Science and Engineering, ISSN: 0975-3397 Vol. 3 No. 7 July 2011, Pp. 2595-2600.

[74] Avani R. Vasant, Sandeep R. Vasant and Dr. G.R. Kulkarni, "Gujarati Character Recognition: The State of the art Comprehensive Survey", Journal of Information, Knowledge and Research in Computer Engineering, Vol. 02, Issue 01.

[75] Rajendra Lambodari and Prof. S. M. Kharad, "Review of Classification Methods for Character Recognition in Neural Network",

International Journal of Electronics Communications and Computer Engineering, Vol. 4, Issue 2, 2013.

[76] Han, J. & Kamber "M. Data mining concept and technique", San Francisco: Morgan Kaufmann Publishers, 2001.

[77] N. Shobha Rani and Smitha Madhukar, "A Decision Tree based font style/size independent Kannada printed character recognition system", International Journal of Science and Research, 2012.

[78] Zhang Ping and Chen Lihui, "A novel feature extraction method and hybrid tree classification for handwritten numeral recognition", 2002.

[79] A. Amin and S. Singh, "Recognition of Hand-Printed Chinese Characters using Decision Trees/Machine Learning C4.5 System.

[80] http://www.freewaregenius.com/2011/11/01/how-to-extract-text-from-images-a-comparison-of-free-ocr-tools/

[81] http://en.wikipedia.org/wiki/Tesseract_(software)

[82] http://en.wikipedia.org/wiki/FreeOCR#User_interfaces

[83] http://www.comptalks.com/convert-images-to-text/

[84] http://www.makeuseof.com/tag/top-5-free-ocr-software-tools-to-convert-your-images-into-text-nb/

[85] http://solutions.weblite.ca/pdfocrx/

[86] http://symmetrica.net/cuneiform-linux/yagf-en.html

[87] http://sourceforge.net/projects/gimagereader/

[88] http://vietocr.sourceforge.net/

[89] https://wiki.gnome.org/Apps/OCRFeeder

[90] https://code.google.com/p/lector/

[91] https://code.google.com/p/lime-ocr/

[92] https://code.google.com/p/tesseract-ocr/wiki/3rdParty

[93] http://www.free-ocr.com/

[94] http://www.i2ocr.com/

[95] http://www.customocr.com/

[96] http://jocr.sourceforge.net/

[97] Shivani Dhiman, A.J. Singh, “Tesseract vs. GOCR A Comparative Study”, International Journal of Recent Technology and Engineering, Vol. 2, Issue 4, Sep. 2013.

[98] C.A.B Mello and R.D Lins, “A Comparative Study on OCR Tools.” Vision Interface '99, Trois-Rivières, Canada, 19-21 May.

[99] http://en.wikipedia.org/wiki/GOCR

[100] SfR Fresh (n.d.). "Member "gocr-0.45/README" of archive gocr-0.45.tar.gz"

[101] http://en.wikipedia.org/wiki/SimpleOCR

[102] http://www.freewaregenius.com/2011/11/01/how-to-extract-text-from-images-a-comparison-of-free-ocr-tools/

[103] http://www.simpleocr.com/Info.asp

[104] http://abbyyindia.com/finereader/professional/about_ocr/whatis_ocr/

[105] http://abbyyindia.com/finereader/professional/features/

[106] Chirag Patel, Atul Patel and Dharmendra Patel, “Optical Character Recognition by Open Source OCR Tool Tesseract: A Case Study”, IJCA, Vol. 55, No.10, October 2012

[107] http://www.ocrtoword.com/

[108] http://products.softsolutionslimited.com/img2ocr/

[109] http://www.a2ia.com/

[110] http://en.wikipedia.org/wiki/Google_Docs

[111] http://www.ocronline.com/

[112] http://www.onlineocr.net/service/about