Multi-font Telugu Text Recognition Using Hidden Markov Models and Akshara Bi-grams

Koteswara Rao Devarapalli[1,2](✉) and Atul Negi[2]

[1] Department of Computer Science and Engineering, Mahatma Gandhi Institute of Technology, Hyderabad 500075, India
dkrao@mgit.ac.in

[2] School of Computer and Information Sciences, University of Hyderabad, Gachibowli, Hyderabad 500046, India

Abstract. Recent advances in the information technology made possible to introduce many Unicode Telugu fonts for the documentation needs of present society. But the recognition of documents printed in a variety of fonts poses new challenges in building Telugu OCR systems. In this paper, we demonstrate multi-font Telugu printed word recognition using implicit segmentation approach that provides segmentation as a by-product of recognition. Our word recognition approach relies on Hidden Markov Models and *akshara* bi-gram language model to recognize word images in terms of *aksharas* (characters). The training set of word images is prepared from document images of popular books and the synthetic document images generated using 8 different Unicode fonts. The testing involves matching the feature vector sequence against sequence of *akshara* HMMs based on bi-grams. The CER and WER of this system are 21% and 37% respectively. The performance of our system is very encouraging.

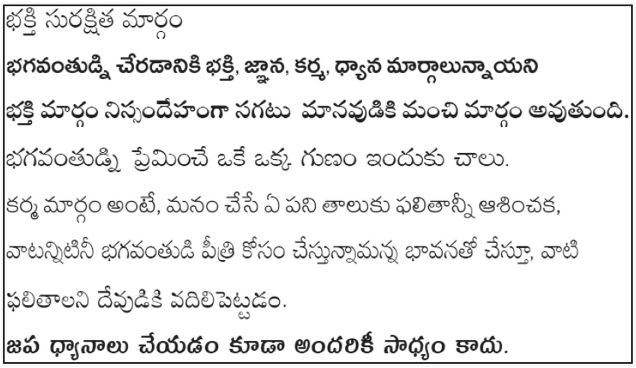
Keywords: Akshara · Bi-gram · DCT · HMM · Telugu OCR · Word recognition

1 Introduction

With advances in information technology, many Telugu fonts are introduced for the documentation needs of present society. Recently developed Unicode fonts for Telugu text editing has made the documentation amazing. A wide variety of font types are in daily use for the documentation needs of print, electronic media, and different organizations. Although there are some contributions made to recognize multi-font Telugu printed text [4,11,13], they target for the recognition of text printed in a few commonly used fonts and those methods can not be directly used for the recognition of text printed in a wide variety of fonts. Hence, the conversion of document images of multi-font Telugu printed text to machine readable form needs suitable novel methods and the necessity of Telugu OCR system to recognize printed text with variations in font has arisen. In addition

S. Mukherjee et al. (Eds.): ICVGIP 2016, LNCS 10481, pp. 243–254, 2017.
https://doi.org/10.1007/978-3-319-68124-5_21

to the complexity of Telugu script, printed text with font variations has become a big challenge to the Telugu OCR systems. The Telugu OCR systems need to be robust enough to recognize arbitrary collection of printed documents with font variations and degradations. In achieving that there are the limitations such as lack of enough examples with natural variations and lack of documentation available about the possible font variations [4]. In Fig. 1, we show an example of multi-font Telugu printed text.


భక్తి సురక్షిత మార్గం

భగవంతుడ్ని చేరడానికి భక్తి, జ్ఞాన, కర్మ, ధ్యాన మార్గాలున్నాయని

భక్తి మార్గం నిస్సందేహంగా సగటు మానవుడికి మంచి మార్గం అవుతుంది.

భగవంతుడ్ని ప్రేమించే ఒకే ఒక్క గుణం ఇందుకు చాలు.

కర్మ మార్గం అంటే, మనం చేసే ఏ పని తాలుకు ఫలితాన్నీ ఆశించక,

వాటన్నిటినీ భగవంతుడి ప్రీతి కోసం చేస్తున్నామన్న భావనతో చేస్తూ, వాటి

ఫలితాలని దేవుడికి వదిలిపెట్టడం.

జప ధ్యానాలు చేయడం కూడా అందరికీ సాధ్యం కాదు.

Fig. 1. Example of multi-font Telugu printed text. The lines are shown from the top to bottom in eight unicode fonts called Mandali, Ramabhadra, Ramaraja, Peddana, Mallanna, Tenali Ramakrishna, Suranna, and Timmana, respectively.

Telugu language is mainly spoken in two south Indian states: Andhra Pradesh and Telangana. Telugu has its own beautiful script, which is written from left-to-right and the text is composed of *aksharas* (characters). An *akshara* is a basic unit of writing that is made up of glyphs. The Telugu orthography has glyphs for vowels, vowel modifiers, consonants, and consonant modifiers. The basic *akshara* set consists of 16 vowels and 36 consonants, though the script is written from left-to-right, vowel modifiers are placed above the consonants and consonant modifiers are written at the below-left, below, and below-right sides of consonants. According to Unicode, the whole *akshara* set comprises of *aksharas* of the form: C^*V^*, where C is a consonant or consonant modifier, and V is a vowel or vowel modifier. In Unicode representation, codes from 0C01 to 0C39, 0C3E to 0C56, and 0C66 to 0C6F are given for Telugu basic *aksharas*, vowel modifiers and numerals respectively. A special character halant (0C4D) is used to define Unicode representation for a consonant modifier. The development of Telugu OCR system involves *akshara* recognition, which is challenging due to large number of like-shaped *aksharas*.

Many of the previous contributions are made for building Telugu OCR based on connected component approach [5,9] and very few of them used language models for recognition. One of the first work appeared in [13] for the recognition of multi-font Telugu characters printed in three popular fonts such as

Harshapriya, Godavari, and Hemalatha in three font sizes. They used connected component approach for segmenting the basic symbols and identified 386 classes of basic symbols that form Telugu basic and compound characters. Based on the type of font used, vowel as well as consonant modifiers may be separated from or connected to a base character that defines the number of basic symbols. They extracted direction features and employed nearest neighbor classifier scheme for recognition.

Another work presented in [11], they explored multi-font Telugu character recognition using histogram of oriented gradients (HOG) features extracted on samples of 359 character classes printed in 15 fonts and SVM classifier is trained. The work presented in [4], they described how to recognize robustly seven major Indic scripts despite font variations and large degradations. They used profile features to represent a word image as a feature sequence and bidirectional long-short term memory (BLSTM) recurrent neural network scheme is used for classification. This neural network scheme allows words to be presented as a sequence of unsegmented features in order to overcome the character segmentation issues in Indian Languages. In [1], they demonstrated omni-font English and Arabic character recognition systems using HMMs. In the literature, many contributions appeared for building character recognition systems based on HMMs in [2,3,8,12].

In our work, we model *akshara* shapes for building Telugu OCR system. Though it has similarity with Hindi OCR based on HMMs. Here, we use 180 *akshara* models to establish the recognition system for Telugu script, whereas Hindi OCR in [8] reported recognition performance based on 50 character models. Our system encompasses the extraction of DCT features to model left-to-right continuous density *akshara* HMMs for representing *akshara* shapes. DCT feature extraction method aims to represent each *akshara* shape as frequency coefficients. The main objective of applying HMMs is to attempt implicit segmentation for Telugu word image recognition that does not need prior segmentation of words into *aksharas* and achieve the segmentation by recognition. The word recognition consists of matching feature vector sequence against sequence of akshara HMMs based on *akshara* bi-gram language model in a maximum likelihood manner.

The paper is organized as follows: we discuss the Hidden Markov Model method in Sect. 2. Language models computation is described in Sect. 3. In Sect. 4, the proposed approach is given. We describe the training of *akshara* HMMs and testing of Telugu words in Sect. 5. Our experiments are described in Sect. 6. Finally, the conclusions are made in Sect. 7.

2 Hidden Markov Models

A Hidden Markov Model is a doubly stochastic process [10]. The Hidden Markov Models are successfully applied for modeling sequence data. Speech recognition is a well known application of HMMs. Similar to speech signal as well as handwritten script, and other real world patterns such as Telugu printed word images written from left-to-right composed of *aksharas* can be modeled as the sequence data that pass through a sequence of states. The states are hidden whereas observation at each point along the word image implies the state. For representing each akshara, we use a 8-state left-right continuous density HMM that has been shown in Fig. 2 with first and last states are empty in order to allow the connectivity between *akshara* models for specifying words. The probability of generating observation is determined by Gaussian mixture components. The state is described by the mean, variance vectors and mixture component weights and it is represented by the frame of an akshara. The system variables such as the frame width, and number of states per *akshara* model are 8-pixels, and 8-states respectively. All these variables are chosen empirically and experimental details are given in Sect. 6.

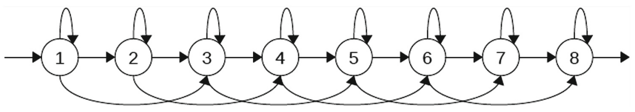


Fig. 2. Structure of *akshara* HMM.

3 Language Models

Language models have widely employed in speech and handwritten recognition applications. The intent of a language model is to predict the sequence of patterns occurring in test data from the frequency of its occurrence in training data. The characters and words are very common patterns in the OCR applications. Further, an n-gram is a sequence of n patterns and the n-gram language model can be used to predict each pattern in the sequence given its $\mathrm{n}-1$ predecessors. Particularly, the bi-gram is a sequence of two patterns, whereas tri-gram is a sequence of three patterns. Both bi-gram and tri-gram language models have widely applied for building character recognition applications in the literature. There are few previous contributions that applied language models for Telugu character recognition. Our work includes the use of *akshara* bi-gram language model to predict the next *akshara* in a sequence of two *aksharas* given its predecessor. The probability of an *akshara* sequence, $P(\hat{A})$ can be approximated as a product of conditional probabilities [15] and is given as follows:

$$\hat{P}(a_1, a_2, \ldots, a_m) \simeq \prod_{i=1}^{m} \hat{P}(a_i | a_{i-n+1}, \ldots, a_{i-1}) for\ n \geq 1 \tag{1}$$

$$\text{where } \hat{P}(a_i | a_{i-n+1}, \ldots, a_{i-1}) = \frac{C(a_{i-n+1}, \ldots, a_i)}{C(a_{i-n+1}, \ldots, a_{i-1})}$$

where C(.) is the count of a given *akshara* sequence in the context. Particularly, the probabilities in an n-gram model can be estimated by counting *akshara* sequences in the ground truth as given in Eq. 1.

4 Proposed Approach

Despite HMMs suitability to model sequential data with variations such hand written text and speech. Our novel approach to recognize multi-font Telugu printed text uses two morphological operations: thinning [6] and re-thickening. Actually the idea of thinning *akshara* patterns of word images then re-thickening to standard thickness is aimed to increase robustness in view of font variation. We use sliding window [2,3] method to obtain frames from Telugu word images. For each frame, we extract a DCT feature vector. The feature vectors are used to model one left-to-right continuous density *akshara* HMM per one *akshara* class [8,12]. We model 180 very frequently occurred *akshara* shapes for Telugu OCR. The Telugu word recognition uses implicit segmentation that involves matching feature vector sequence of testing word image against sequence of *akshara* HMMs based on *akshara* bi-gram language model in a maximum likelihood manner.

4.1 Preprocessing

The main role of preprocessing is to convert the input Telugu document images into data of desirable form in such a way that it can be readily used for extracting useful features. Different preprocessing tasks such as noise removal, binarization, line segmentation, and word segmentation are used in this work. First, we use 3×3 median filter for the removal of commonly occurred kind of noise called salt-and-pepper noise. The binarization is performed using Otsu's method to convert Gray scale document images into binary images. The horizontal projection profiles (HPPs) and vertical projection profiles (VPPs) of binary document images are used to segment lines and words respectively. We also exploit two morphological operations: thinning and re-thickening of Telugu word images to bring them into standard form despite font variation.

4.2 Thinning

We intend to perform thinning operation on Telugu word images for reducing *akshara* patterns as much as possible without changing their general shape. Thinning reduces an image while keeping its topological and geometric properties unchanged. Actually thinning involves either deleting or retaining a foreground pixel based on the configuration of pixels in its local neighborhood. We perform thinning operation based on standard algorithms for converting Telugu word images with font variations into standard form [6,14]. The outcome of thinning is a skeleton which is a collection of thin arcs and curves. In our work, thinning reduces each *akshara* pattern of the given word into a skeleton through deleting layers of pixels on its boundary.

4.3 Re-thickening

The re-thickening operation consists of changing the width of strokes of *aksharas* uniformly despite font variations. That is, it involves increasing the width of

1	1	1
1	1	1
1	1	1

Fig. 3. Structuring element used for dilation operation.

strokes of a pattern using dilation operation. Dilation is the basic operation to add pixels on the boundary of a given image. Since thinning operation causes an image to loss successive layers of pixels on its boundary, the image can regain pixels on its boundary to standard thickness through re-thickening. We use the standard structuring element shown in Fig. 3 to perform dilation in order to uniformly increase *akshara* stroke width in such a way that the *akshara* has standard thickness. The sample multi-font Telugu printed word images and their equivalents after performing the morphological operations: thinning and re-thickening are shown in Fig. 4.

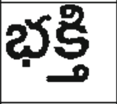
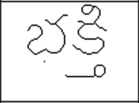
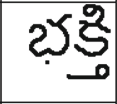
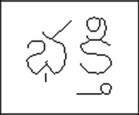
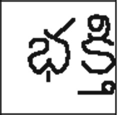
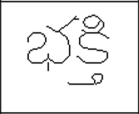
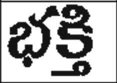
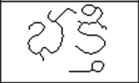
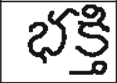
S.No	A	B	C
1			
2			
3			
4			
5			

Fig. 4. Examples of multi-font word images, their thinned and re-thickened equivalents are given in A, B and C columns respectively.

4.4 Feature Extraction

We intend to use discrete cosine transform (DCT) to extract significant information from Telugu word images in order to represent *akshara* shapes. The DCT is a real-valued discrete sinusoidal unitary transform. It is a widely used mathematical tool for converting an image from the spatial domain to frequency domain through projecting onto a set of basis functions. Thus the DCT allows us to extract elementary frequency components as features from an image without changing its information content.

$$Dpq = \alpha_p \alpha_q \sum_{k=0}^{M-1} \sum_{l=0}^{N-1} I_{kl} \cos \frac{\pi(2k+1)p}{2M} \cos \frac{\pi(2l+1)q}{2N} \tag{2}$$

$$\text{where } \alpha_p = \sqrt{1/M} \ if \ p = 0, \sqrt{2/M} \ if \ p \neq 0$$

$$\alpha_q = \sqrt{1/N} \ if \ q = 0, \sqrt{2/N} \ if \ q \neq 0$$

where I and D are the frame of an *akshara* and its DCT transform respectively. During feature extraction, we use sliding widow along word images in writing order to obtain frames. In Fig. 5(a), all operations performed toward extraction of frames are shown. The width and height of our sliding window is 8 and 80 pixels respectively with no overlapping between windows. So the feature vector is of 640-dimensional. For each frame, we perform DCT transform and then map the resultant matrix into a 1-D feature vector. We make use of feature vectors extracted from training set for modeling *akshara* HMMs, while feature vectors obtained from test set are matched against sequence of *akshara* HMMs based on bi-grams to recognize *aksharas* and words.

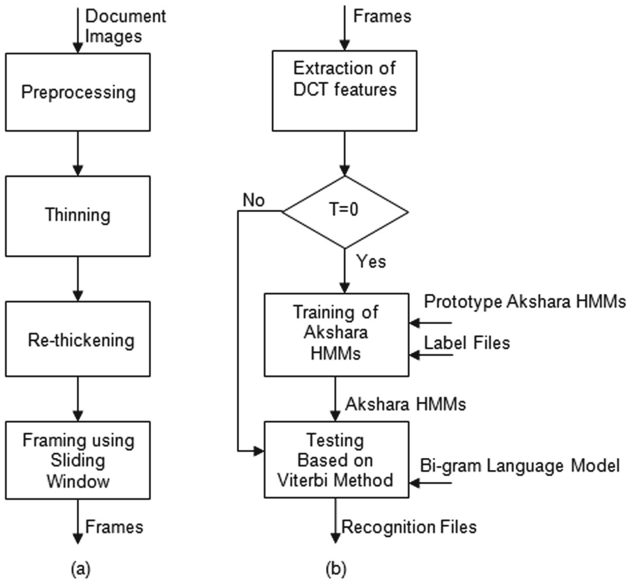


Fig. 5. The structure of multi-font Telugu printed word recognition system: (a) framing, (b) training and testing.

5 Training and Testing Processes

The HMM tool kit (HTK) development environment is used for training of *akshara* HMMs and to test Telugu word images [15]. The training is performed using the HERest tool of HTK that is based on Baum-Welch algorithm. Whereas the HVite tool, which is based on Viterbi algorithm, is used for testing. Both the processes are shown in Fig. 5 and the control variable T is used to choose between them.

5.1 Training Process

The training process involves various tasks such as prototype *akshara* HMM specification, initialization of *akshara* HMMs, parameter estimation and model refinement. The input to the training process are feature vectors extracted from training set, prototype *akshara* HMMs and label files. During training, once the feature vectors extracted from training set are ready, we use HCompV tool to compute global variance. Then, the states of prototype model are set in such a way that mean is initialized to vector of zeros, while the variance is set to global variance. Thus all required initial models are defined based on the prototype *akshara* HMM. Finally we employ embedded training approach for parameter estimation that involves simultaneous estimation of mean, variance, and Gaussian component mixture weights of all states of all *akshara* models appeared in the training set. The model refinement consists of modifying the models and then re-estimating their parameters.

5.2 Testing Process

The whole objective of the system can be fulfilled through testing process. The testing process involves recognition of words and *aksharas* through matching feature vector sequences against *akshara* HMMs. The standard Viterbi decoder is used to match feature vector sequence against *akshara* HMMs, whereas the HVite decoder can be used for matching feature vector sequence against sequence of *akshara* HMMs (word). The HVite decoder is based on both Viterbi and token passing algorithm [15]. During testing, the decoder needs feature vectors extracted on testing set of word images, trained *akshara* HMMs, and bi-gram language model. The testing process consists of matching the feature vector sequence against sequence of akshara HMMs based on bi-gram language model in order to find the maximum likelihood *akshara* HMM sequence that describes the recognized word in the system.

6 Experiments

Our training and testing data are 7000 word images segmented from 35 document images of books printed in 4 widely used Telugu fonts. Data also includes 900 word images segmented from synthetic document images generated in 8 Unicode

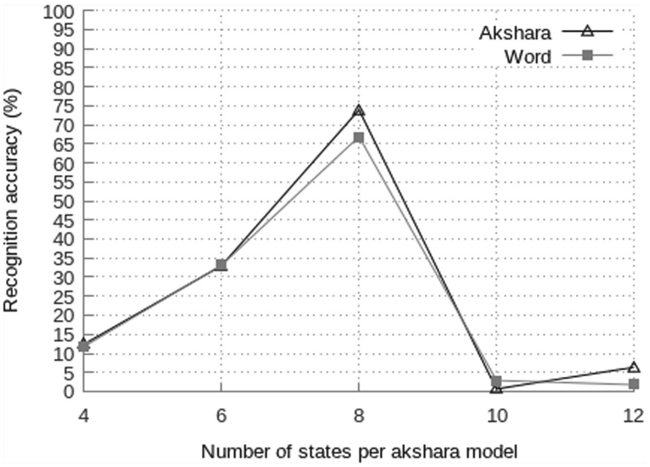


Fig. 6. *Akshara* and word recognition percent accuracies as the functions of states.

fonts. In our experiment, we extract DCT features from 4950 Telugu word images to train 180 *akshara* models. Whereas the *akshara* bi-grams are computed for 180 *aksharas* on the text corpus of 10,000 words that are most frequently used in the Telugu language. We use 850 word images to validate our approach, whereas the test set of 700 word images is employed to establish the performance.

Validation of the approach is performed to evaluate how our recognition system is meeting the requirements and to fix system parameters such as sliding window width and number of states per model. The standard metric called *percent accuracy* is used to demonstrate how the system recognition capability varies with number of states per model. In order to fix the number of states per *akshara* model, we define *akshara* models with a different number of states such as 4, 6, 8, 10 and 12. Then, the models are trained using feature vectors extracted from fixed width frames of 8-pixels and both the *akshara* as well as word recognition capabilities are evaluated. We empirically observe good recognition capability with *akshara* models of 8-states and frame width of 8-pixels. The percent accuracy curves of *akshara* and word recognition results are shown in Fig. 6. Moreover, the character error rate (CER), and word error rate (WER) metrics are used for representing cost to the user of a given recognition system [7] in the literature, so we intend to report the system performance using the CER, and WER. The CER is defined as follows:

$$CER = \frac{D + I + S}{T} \tag{3}$$

where D, I, S, and T denote deletion, insertion, substitution errors and T represents the total number of characters in the ground truth respectively. For each word of the test set of 700 word images, feature vector sequence computed is matched against sequence of *akshara* HMMs based on bi-gram language model

Table 1. Telugu *akshara* and word error rates

Type of preprocessing	CER	WER
Without preprocessing	35	50
After thinning	35	56
After thinning and re-thickening	21	37

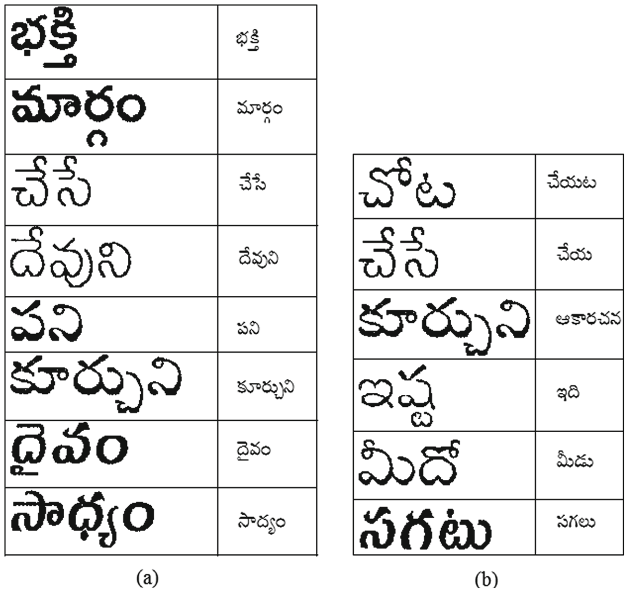

భక్తి
మార్గం
చేసే
దేవుని
పని
కూర్చుని
దైవం
సాధ్యం
(a)
చేయుట
చేయు
ఆకారచన
ఇది
మీడు
సగలు
(b)

Fig. 7. Example multi-font Telugu printed word images: (a) successfully recognized words with their equivalent text, (b) word images recognized with some *akshara* errors.

to find the maximum likelihood sequence (word). Finally the error rates are estimated. The CER and WER of our system are given in Table 1. The error rates are compared to demonstrate the significance of thinning and re-thickening. The merit of using bi-gram language model is that it does not need to specify the sequences of *akshara* HMMs (words). This system does not use lexicon either during recognition or at post-processing level. The lexicon along with bi-grams can be useful to reduce the error rates [8] further. In Fig. 7(a), a sample of successfully recognized multi-font Telugu word images are shown along with their equivalent text. Thus the given Telugu words are recognized through implicit segmentation based on HMMs. This approach makes use of bi-gram language model and does not need prior segmentation of words into *aksharas* before recognition. We also show example word images that are recognized with some *akshara* errors in Fig. 7(b).

7 Conclusion

We proposed a novel approach for multi-font Telugu printed word recognition using Hidden Markov Models. In addition to the HMMs suitability to model sequential data with variations, this approach further uses the thinning and re-thickening operations to obtain *aksharas* of standard stroke width in view of font variation and extracts DCT features through sliding window for modeling the shape of *aksharas* using HMMs. The merit of implicit segmentation for Telugu word recognition is to achieve segmentation by recognition. The presented system does not rely on lexicon, but we intend to use lexicon to reduce errors. Our future work includes applying this approach for recognition of line images based on tri-grams and using a combination of classification approaches for improving the performance.

References

1. Bazzi, I., Schwartz, R., Makhoul, J.: An omnifont open-vocabulary OCR system for English and Arabic. IEEE Trans. Pattern Anal. Mach. Intell. **21**(6), 495–504 (1999)
2. Elms, A., Procter, S., Illingworth, J.: The advantage of using an HMM-based approach for faxed word recognition. Int. J. Doc. Anal. Recogn. **1**(1), 18–36 (1998)
3. Khorsheed, M.S.: Offline recognition of omnifont Arabic text using the HMM toolkit (HTK). Pattern Recogn. Lett. **28**(12), 1563–1571 (2007)
4. Krishnan, P., Sankaran, N., Singh, A.K., Jawahar, C.V.: Towards a robust OCR system for Indic scripts. In: 2014 11th IAPR International Workshop on Document Analysis Systems (DAS), pp. 141–145, April 2014
5. Kumar, P.P., Bhagvati, C., Negi, A., Agarwal, A., Deekshatulu, B.L.: Towards improving the accuracy of Telugu OCR systems. In: ICDAR, pp. 910–914. IEEE Computer Society (2011)
6. Lam, L., Lee, S.-W., Suen, C.: Thinning methodologies-a comprehensive survey. IEEE Trans. Pattern Anal. Mach. Intell. **14**(9), 869–885 (1992)
7. Natarajan, P., Lu, Z., Schwartz, R., Bazzi, I., Makhoul, J.: Multilingual machine printed OCR. Int. J. Pattern Recogn. Artif. Intell. **15**(01), 43–63 (2001)
8. Natarajan, P., MacRostie, E., Decerbo, M.: The BBN byblos Hindi OCR system. In: Govindaraju, V., Setlur, S. (eds.) Guide to OCR for Indic Scripts. Advances in pattern recognition, pp. 173–180. Springer, London (2010). doi:10.1007/978-1-84800-330-9_9
9. Negi, A., Bhagvati, C., Krishna, B.: An OCR system for Telugu. In: ICDAR, pp. 1110–1114. IEEE Computer Society (2001)
10. Rabiner, L.: A tutorial on hidden Markov models and selected applications in speech recognition. Proc. IEEE **77**(2), 257–286 (1989)
11. Rasagna, V., Jinesh, K.J., Jawahar, C.V.: On multifont character classification in Telugu. In: Singh, C., Singh Lehal, G., Sengupta, J., Sharma, D.V., Goyal, V. (eds.) ICISIL 2011. CCIS, vol. 139, pp. 86–91. Springer, Heidelberg (2011). doi:10.1007/978-3-642-19403-0_14
12. Roy, P., Roy, S., Pal, U.: Multi-oriented text recognition in graphical documents using HMM. In: 2014 11th IAPR International Workshop on Document Analysis Systems (DAS), pp. 136–140, April 2014

13. Vasantha Lakshmi, C., Patvardhan, C.: A multi-font OCR system for printed Telugu text. In: 2002 Proceedings of Language Engineering Conference, pp. 7–17, December 2002
14. Wu, Y., Shivakumara, P., Wei, W., Lu, T., Pal, U.: A new ring radius transform-based thinning method for multi-oriented video characters. IJDAR **18**(2), 137–151 (2015)
15. Young, S., Evermann, G., Gales, M., Hain, T., Kershaw, D., Liu, X.A., Moore, G., Odell, J., Ollason, D., Povey, D., Valtchev, V., Woodland, P.: The HTK Book (for HTK Version 3.4). Cambridge University Engineering Department (2006)