

Computationally efficient handwritten Telugu text recognition

Buddaraju Revathi¹, M. V. D. Prasad¹, Naveen Kishore Gattim²

¹Department of Electrical and Computer Engineering, Koneru Lakshmaiah Education Foundation, Vaddeswaram, India

²Department of Electrical and Computer Engineering, Sasi Institute of Technology and Engineering, Tadepalligudem, India

Article Info

Article history:

Received Jan 7, 2024

Revised Feb 21, 2024

Accepted Mar 9, 2024

Keywords:

Character recognition

Feature extraction

Inception

Neural network

Orthographic features

ABSTRACT

Optical character recognition (OCR) for regional languages is difficult due to their complex orthographic structure, lack of dataset resources, a greater number of characters and similarity in structure between characters. Telugu is popular language in states of Andhra and Telangana. Telugu exhibits distinct separation between characters within a word, making a character-level dataset sufficient. With a smaller dataset, we can effectively recognize more words. However, challenges arise during the training of compound characters, which are combinations of vowels and consonants. These are considered as two or more characters based on associated vattus and dheerghams with the base character. To address this challenge, each compound character is encoded into a numerical value and used as input during training, with subsequent retrieval during recognition. The segmentation issue arises from overlapping characters caused by varying handwritten styles. For handling segmentation issues at the character level arising from handwritten styles, we have proposed an algorithm based on the language's features. To enhance word-level accuracy a dictionary-based model was devised. A neural network utilizing the inception module is employed for feature extraction at various scales, achieving word-level accuracy rates of 78% with fewer trainable parameters.

This is an open access article under the [CC BY-SA](#) license.



Corresponding Author:

Buddaraju Revathi

Department of Electronics and Computer Engineering, Koneru Lakshmaiah Education Foundation

Vaddeswaram, Guntur, Andhra Pradesh, India

Email: buddaraju.revathi@gmail.com

1. INTRODUCTION

Automation plays a vigorous role in the digital world. Data entry came into existence to extract handwritten text from application forms, bank cheques, and scanned documents to maintain their record. Data entry is an important and tough task to perform manually. We can take a photo of the document and store it, but we cannot make filtering operations like sort, find on a particular field of data in the scanned document. Data entry engineer will manually enter the data in the documents onto the computer that can be edited as per the need. Automation of text in images is a current area of interest in pattern recognition which is a challenging task and the process used for this is called optical character recognition (OCR).

Regional languages are becoming prominent. OCR for Indic languages is gaining importance to automate the text in regional languages for postal services, banks, and government forms [1]. Searching the content in regional languages is gaining interest now a days, but the search engine needs a translator to search the content which is in regional languages. So, there is importance of translator. If OCR for Indic languages is developed it provides support in preparation of dataset for translator. Using OCR, we can also preserve historical documents which contain valuable information of ancient history [2]. OCR is implemented in five

phases [3]. The first phase is page detection. Scanning photo for document may contains different background, so before detecting text in the image we need to detect the page borders and remove the unnecessary background details as well as correct the inclinations if any [4].

The second phase consists of detecting the lines and words in the image. Spaces between the lines will be maintained while writing text in any language. Taking these gaps into account we can separate the lines in the image [5], [6]. There will be a gap between the words in any line which will help to detect the words in the lines. Most popularly used techniques are horizontal and vertical histograms, Sobel, Canny edge detection, and Perwitt.

Third phase is character segmentation, a crucial step in OCR [7], [8]. This phase depends on language. If the language has gaps between the characters, then character segmentation becomes easy. If there is no gap between the characters, we can use word level training or segment the word into characters based on the stroke of writing, which is difficult. Now a days character level segmentation is a demanding task in OCR [9], [10]. Fourth phase is training and testing. Text recognition for regional languages is difficult as a character might be single or compound. Challenges in developing Telugu OCR are primarily segmentation of characters and feature extraction due to a greater number of characters with complex orthographic structure and similarity between them. Examples are య, ప; స, శ; ష, ష; ఠ, ఠ.

Researchers have been focusing their efforts on developing OCR systems for native languages. One of the primary challenges encountered in this endeavor, particularly with limited datasets, is character segmentation. Additionally, selecting a suitable model capable of effectively extracting complex features inherent to the native language poses another significant hurdle in the development process. For the segmentation of text in the images into words and characters is difficult for Indic languages. Mamatha and Srikantamurthy [11] worked on Kannada character segmentation using morphological operations, which efficiently extracts the lines in the text. Later applied projection profiles for detecting characters form words that are in text lines. Inkeaw *et al.* [12] worked on segmentation of characters for Lanna Dhamma that have different writing style, touching characters or broken characters using GPCSeg. Garain and Chaudhuri [10] worked on segmentation of Bangla and Devanagari scripts. Touching characters are separated using fuzzy multi factorial analysis.

To tackle the second problem, advancement of deep learning techniques attracted the research to incorporate them in OCR. Huang *et al.* [13] worked on convolution neural network (CNN) along with sliding window to improve character recognition accuracy in typical scene images. MSERs are used to improve the quality of detection. Mathew *et al.* [14] recognized text in the scene images that are in Telugu, Malayalam, and Devanagari by using CNN-recurrent neural network (RNN) model which was used for training images from end-to-end. This model recognizes text in the images by detecting the words from them. Prakash *et al.* [15] worked on Telugu OCR for printed text. Connected components are used for segmenting images at character level. CNN is used for feature extraction of base character, vattu and gunintham. Adam optimizer is employed. CNN is used as a classifier. Tong *et al.* [16] worked on MA-CRNN for identification of complex text lines in scene images that are in Chinese. Bidirectional long short-term memory (Bi-LSTM) with attention mechanism is used for predicting the sequences, which increased recognition accuracy of long lines. Weng and Xia [17] proposed an OCR for handwritten character recognition on mobiles. They constructed a Shui dataset. Light weight CNN structure is used for feature extraction and classification on mobiles. Zhao *et al.* [18] worked on OCR for Kannada and English scene text detection. It is based on VGG-Net with less units of processing in convolutional layer. Different strategies for initialization are used for achieving better accuracy rates.

The segmentation algorithm developed must account for language-specific ligatures, particularly due to the presence of compound characters in native languages. However, the existing algorithms fail to adequately segment Telugu characters, especially in cases of overlapping and compound characters. To address this gap, we have devised a character segmentation algorithm specifically tailored for Telugu, considering the lack of similar work in this language. While CNN architectures such as AlexNET, VGG, and inception have been widely utilized for text recognition in images [19], [20]. Previous research has often encountered challenges with long training times for native languages [16], [18]. In response, we have developed a lightweight model optimized for accurately recognizing Telugu language features, leveraging the capabilities of inception V1.

2. INCEPTION BASED OPTICAL CHARACTER RECOGNITION

2.1. Pre-processing and word recognition

Initially the test image is transformed into grey scale and page borders are detected to remove skew in the image. Bilateral filters prove highly effective in reducing noise and smoothing edges in grayscale

images owing to their distinctive filtering technique. Adaptive thresholding differentiates the page with background of the image. Mathematically threshold value is expressed by (1),

$$\text{Th}(r, s) = w(r, s) - \text{cstparam} \quad (1)$$

where $\text{Th}(r, s)$ is threshold value, $w(r, s)$ is weighted average and cstparam is constant parameter.

Median blur is applied to remove details inside the page, to detect the page edges. Later, canny edge detection detects nearby edges of the page. To remove unwanted edges morphological processing is used. From the detected edges of page, height and width of the page are calculated to create contours. So, the resultant image does not contain unwanted details outside the page.

For detecting the words in the page, initially remove the noise in image by applying gaussian filtering. This is done by convolving with gaussian kernel of size (5×5) . General expression for gaussian kernel filter which has size of $(2n+1) \times (2n+1)$ is given by (2) [21].

$$T_{ij} = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{(i-(n+1))^2 + (j-(n+1))^2}{2\sigma^2}\right); 1 \leq i, j \leq (n+1) \quad (2)$$

Sobel operator, which consists of discrete differentiation is used to find intensity gradients. The direction and gradient for the edges are computed by (3).

$$G = \sqrt{G_i^2 + G_j^2} \text{ and } \theta = a \tan 2(G_j, G_i) \quad (3)$$

where G_i, G_j represents horizontal and vertical derivatives.

Non-maximum suppression is applied to merge all detections that belong to same word. Usually, the result of non-maximum suppression will not preserve edges correctly, so strong and weak edges are detected by applying double thresholding to exactly mark boundaries of the words. Strong edges are preserved. Edges which are weak but linked to strong edges are retained and preserved. Later bounding rectangles are created for the detected edges.

2.2. Character segmentation and feature extraction

This phase is exceptionally significant because the precision of OCR relies heavily on the character segmentation algorithm utilized. Handwritten text presents a challenge due to the possibility of characters overlapping with their neighboring characters, making accurate segmentation difficult. If character segmentation is not precise, it significantly impairs the OCR's performance. To tackle this issue, we have introduced an algorithm specifically designed for segmenting handwritten Telugu text. Figure 1 gives a process of character segmentation with an input as one bounding box. This process is repeated for all bounding boxes in the page to obtain characters in the page.

CNNs are powerful feature extractors to capture significant features in an image by using convolution operation [22]. We have taken character dataset from IEEE data port and trained it using inception V1 model which consists of concatenated convolutional layers with variations in kernel sizes and max pooling layers to capture the features of input characters at different scales due to their complex structure. Consider an image of region u which is of size $y \times z$ and a convolutional filter r , without bias term, then function g would be given by (4).

$$g(u, r) = u * r = \sum_{y,z} r_{y,z} u_{y,z} \quad (4)$$

So, the function g passes through origin and model gets over-fit for points only passing through origin. With addition of bias the expression changes as given by (5).

$$g(u, r, b) = b + u * r = b + \sum_{y,z} r_{y,z} u_{y,z} \quad (5)$$

So now the model added with bias will be flexible to train and avoids over-fit, and the above expression closely resembles the functionality of human neurons, so usually bias term is added for the resultant of convolution operation. We have used rectified linear units (ReLU) activation for inception and He-normal initialization for initializing weights of neurons, which have faster convergence. When activation is ReLU the output is given by (6),

$$Y = w_1s_1 + w_2s_2 + \dots + w_n s_n \quad (6)$$

where w_1, w_2, \dots, w_n are the weights of n neurons, s_1, s_2, \dots, s_n are n inputs and Y is the output. He-normal initializes the weights by taking them from distribution, whose variance is given by (7),

$$Var(w_i) = \frac{2}{fan-inputs} \tag{7}$$

where fan-inputs represent the number of weight tensor input units.

The resultant output is flattened and given as an input to the dropout layer. In dropout we drop connections that are present between the neurons, thereby reducing the chances of over-fitting. Now the segments of the test page which are Telugu characters are given to the trained model for prediction. Later SoftMax is applied to classify images, which uses the expression to generate discrete probabilities of all classes between 0 and 1 given by (8),

$$\sigma(s_i) = \frac{e^{s_i}}{\sum_{h=1}^k e^{s_h}} \tag{8}$$

where k represents total classes, s represents a vector corresponding to input, $\sigma(s_i)$ represents probability of output class. The highest probability character is the predicted output.

Adam optimizer is used for adjusting weights of the network. During training we need an optimizer to ensure the loss function is minimized by revising the model weights and moving in the correct direction. Adam optimizer, which updates the weights as in (9),

$$r_{t+1} = r_t - \frac{\eta}{\sqrt{\hat{u}_t + \epsilon}} \hat{p}_t \tag{9}$$

where \hat{p}_t -first moment corrected bias estimate, \hat{u}_t -second moment corrected bias estimate, $\epsilon = (10 \times \exp(-8))$, η -learning rate, r_{t+1} - weight at $t+1$ time instant, r_t - weight at t time instant.

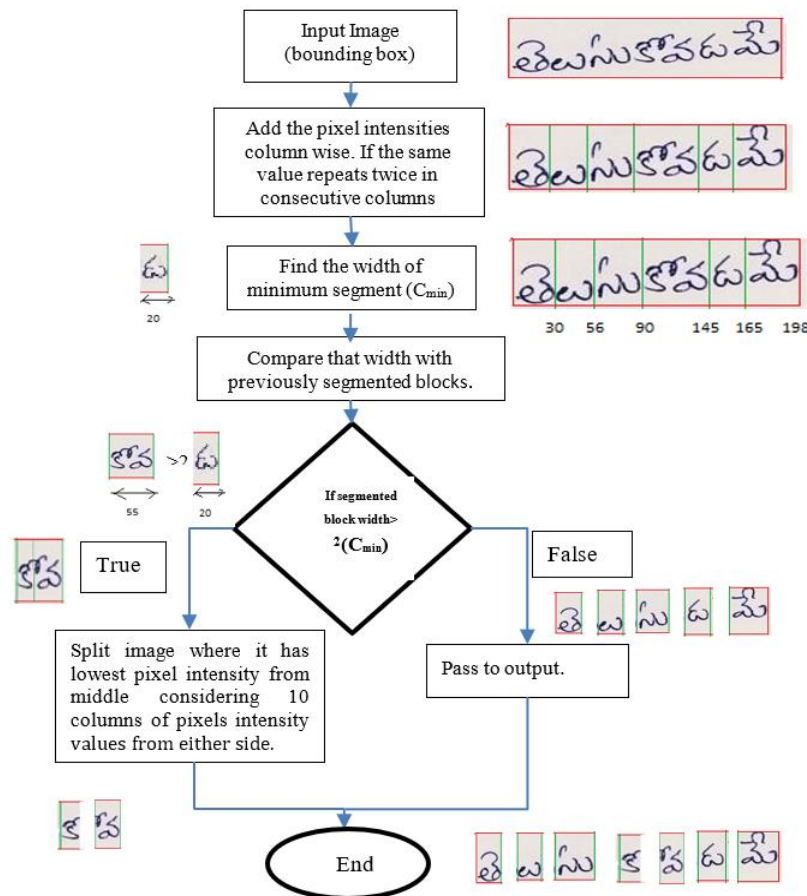


Figure 1. Flow chart of character segmentation

Previously, many researchers relied on support vector machines (SVM) classifiers and CNN variations for native language OCR tasks [15], [16], [18]. However, SVM’s recognition rate was found to be less significant, while other models, being deep, incurred extensive training times. In response, we opted for inception V1 models, modifying layers to suit our needs. Selection of layers was based on careful observation of accuracy and loss values across different layer and filter size variations. The proposed inception model depicted in Figure 2 consumes fewer parameters yet achieves accuracy rates comparable to other CNN models.

Inception V1 is used for extracting structures of the image. The neural network model is in Figure 2. Inception V1 achieves efficiency by strategically utilizing 1×1 convolutions, reducing the overall parameter count significantly. Furthermore, the incorporation of parallel convolutions enables the network to capture features across diverse scales, thereby enhancing its proficiency in recognition tasks.

Inception module emphasizes broader patterns, empowering the network to discern difficult structures within the data [20]. The 1×1 convolution utilizes a small kernel size, allowing the network to grasp complicated details and reduce input channels, thereby enhancing parameter efficiency. On the other hand, the 3×3 convolution captures medium-sized features, striking a balance between local and global information. This adaptability enables the model to effectively recognize intermediate-level patterns. Meanwhile, the 5×5 kernel operation captures larger, complex features, concentrating on broader patterns. This equips the network with the ability to identify difficult structures within the data. These features of inception V1 makes it suitable to extract features of Telugu characters.

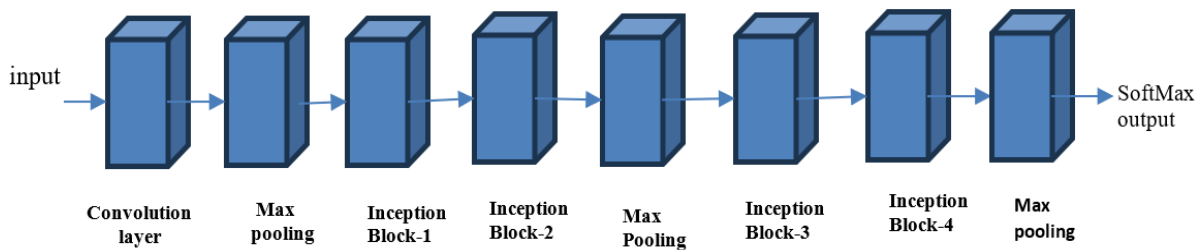


Figure 2. Neural network model

We have incorporated dictionary model to improve word accuracy of OCR. The OCR receives scanned handwritten text as input. After training the neural network with input images, the model identifies characters within the words. However, inaccuracies can arise due to similar patterns in characters, human errors, or OCR misrecognitions. To tackle these issues, we have designed a dictionary model. In this approach, detected words undergo processing through this algorithm, which addresses problems related to segmentation and variations in handwritten styles. The process within the dictionary model is illustrated in Figure 3. The suggested inception model demonstrates proficiency in extracting Telugu language features within a shorter training period. Moreover, the incorporation of a dictionary model effectively addresses the correction of similar characters such as య, మ; స, న; ధ, ధ; ర, ర, thereby augmenting the overall accuracy of the system.

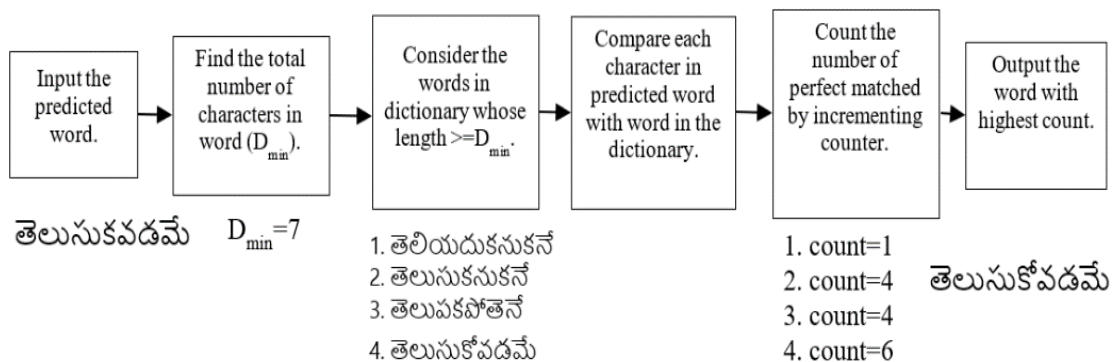


Figure 3. Dictionary model

3. RESULTS

The dataset is taken from IEEE data port [23]. All images are scaled to 64×64 pixels of constant height and width. Later they are converted into grey scale. Two models, VGG-19 and inception V1, are trained using 11,602 images, and 2,565 images are utilized for testing. Both models undergo 25,000 training steps. Figure 4 shows the plots of VGG-19. Figure 4(a) illustrates the accuracy plot, and Figure 4(b) displays the loss plot for VGG-19. VGG-19 attains an accuracy of 89.5% with a validation loss of 10.5%. The plots of Inception V1 are shown in Figure 5. Inception V1 accuracy plot is shown in Figure 5(a) and loss plot in Figure 5(b). Inception V1 has achieved an accuracy of 87% with validation loss of 13%.

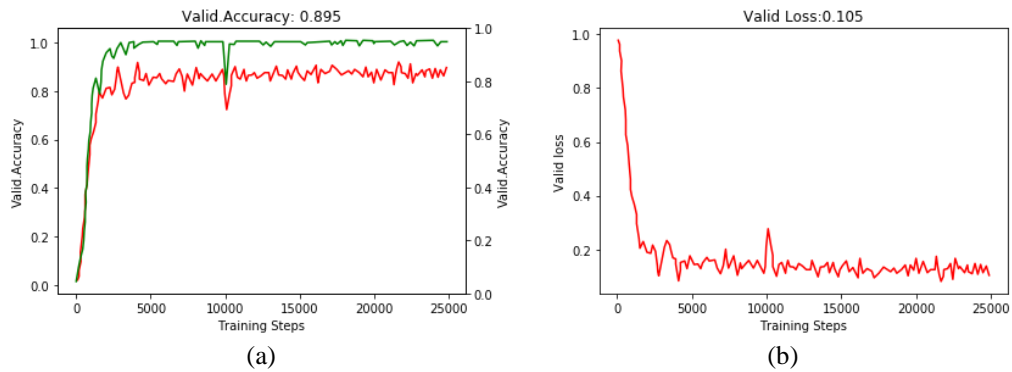


Figure 4. Plots of VGG-19 (a) accuracy graph of VGG-19 and (b) loss graph of VGG-19

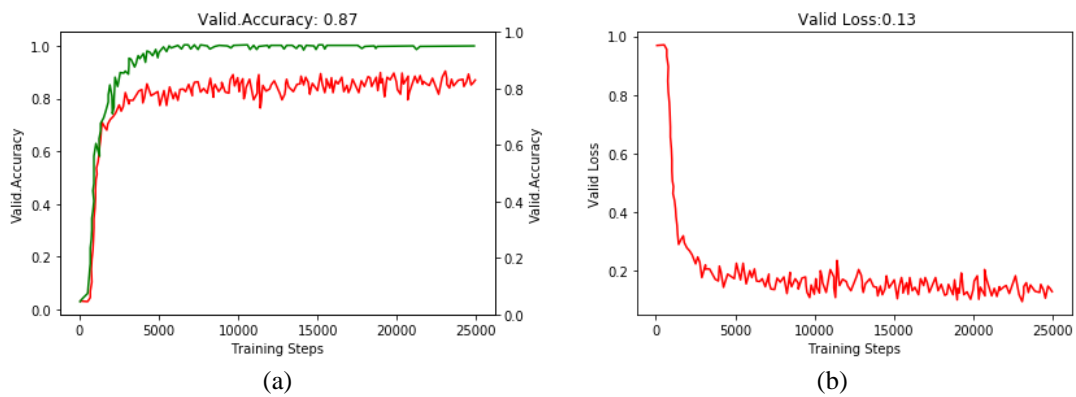


Figure 5. Plots of inception V1 (a) accuracy graph of inception V1 and (b) loss graph of inception V1

The model with more accuracy and less loss is considered better. Accordingly, VGG model attained the highest accuracy among the two models. However, it comes with a drawback that the total number of parameters is significantly high, leading to extended training times compared to other models. The inception model, while approaching the accuracy levels of VGG, managed to achieve nearer results with fewer parameters.

We have developed a dictionary and written 1,000 words and given them as input to the OCR. Later calculated the word recognition rates with and without using dictionary. Inclusion of this model enhanced the recognition rates. Accuracy rates for models, both with and without a dictionary, are presented in Table 1. In this table, character recognition rate (CRR), and word recognition rate (WRR).

Table 1. Comparison of inception V1 and VGG-19

Model	Parameters	Accuracy without dictionary		Accuracy with dictionary
		CRR	WRR	WRR
Inception V1	56,31,016	87%	76%	78%
VGG-19	1,44,02,816	89.5%	78%	79.5%

Inception V1 has 56,31,016 trainable parameters and has 87% CRR and 76% WRR without dictionary. The dictionary model enhanced the WRR to 78%. VGG-19 used 1,44,02,816 trainable parameters and achieved 89.5% CRR, 78% WRR without dictionary. The dictionary model enhanced WRR to 79.5%. Inception V1 requires less training time than VGG-19 as it requires less trainable parameters and achieved accuracy almost nearer to VGG-19. The decision between VGG-19 and inception V1 depends on the unique requirements of the task and the available computational resources.

Figure 6 shows the comparison of VGG-19 and Inception V1. The comparison of validation loss for VGG-19 and inception V1 is shown in Figure 6(a) and the parameters comparison is in Figure 6(b). Loss plots play a vital role in machine learning model development, aiding in understanding, improving, and effectively communicating model performance. Loss plot shows that VGG-19 has less validation loss compared to inception V1. From parameter plot it is clear that VGG-19 uses more trainable parameters and requires more computational requirements than inception V1. The training time required for VGG-19 is more compared to inception V1.

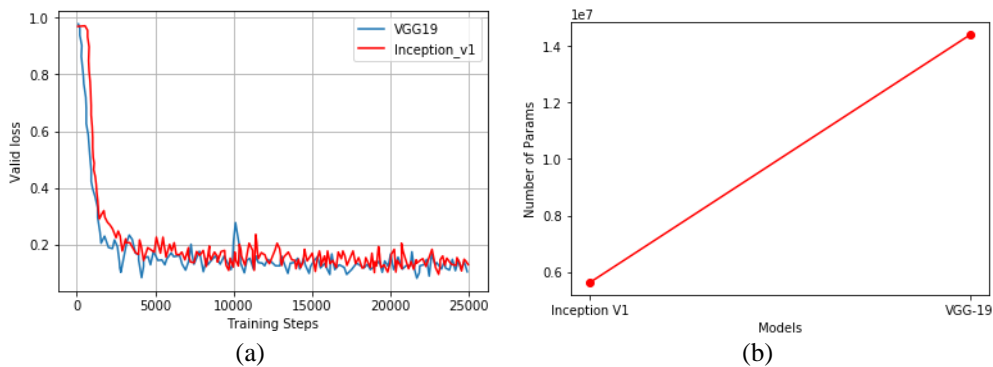


Figure 6. Comparison plots of VGG-19 and inception V1 (a) loss plots V1 and (b) parameters plot

Performance comparison of VGG-19 and Inception V1 is shown in Figure 7. Character and word recognition rates comparison is in Figure 7(a). The CRR are WRR for both the models are significant. The WRR comparison with and without dictionary model is shown in Figure 7(b). VGG-19 achieved a CRR of 89.5%, whereas inception V1 achieved 87%. In terms of WRR, VGG-19 scored 78%, while inception achieved 76%. Dictionary models eliminated some of the problems with similar structured pattern recognition and enhanced the recognition rate of VGG-19 to 79.5% and inception V1 to 78%.

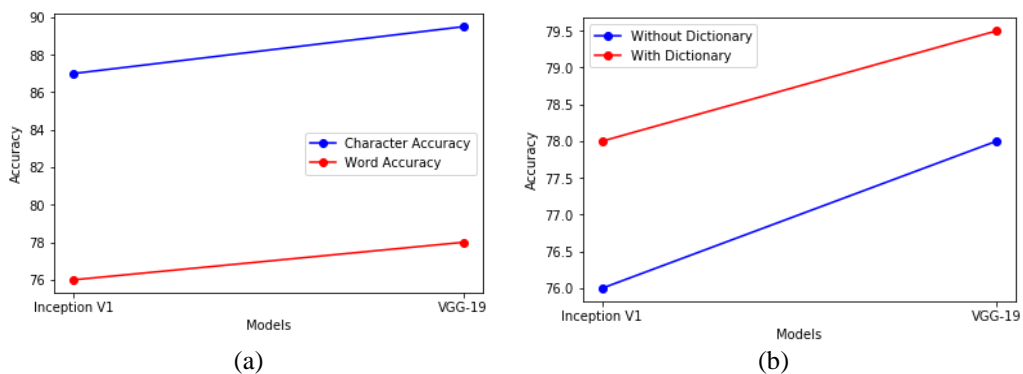


Figure 7. Performance comparison of VGG-19 and inception V1 (a) recognition rates without dictionary V1 and (b) word recognition rates with and without dictionary

The above plots show that inception V1 can attain accuracy rate nearer to VGG with short training period. Some researchers have delved into the development of Telugu OCR systems [14], [24], [25]. Table 2 illustrates a comparison of various techniques employed for Telugu OCR. It's apparent from the table that

limited efforts have been devoted to handwritten text recognition, primarily due to challenges in segmenting words into characters. However, the developed segmentation algorithm adeptly addresses this issue, accurately segmenting words into characters. Notably, the inception V1 model has achieved significantly high recognition rates compared to alternative approaches. The proposed model holds promise for automating Telugu text in government forms and for educational purposes.

Table 2. Comparison of accuracy rates for Telugu OCR

Name of the author	Type of text	Technique	Accuracy
Rani [24]	Handwritten Telugu characters	SVM classifier	80%
Rao [25]	Handwritten Telugu text	Hidden Markov models (HMMs), Akshara models and Akshara Bigram language models	74%
Mathew [14]	Scene Telugu text detection	Hybrid CNN-RNN (CRR)	86.2%
		Hybrid CNN-RNN (WRR)	57.2%
Proposed	Handwritten Telugu text	CNN, Inception V1 (CRR)	87%
		CNN, Inception V1 (WRR)	78%

4. CONCLUSION

Automating handwritten Telugu text remains challenging due to segmentation issues and the complexity of extracting features from similarly structured numerous characters. Previous OCR techniques for handwritten text have not achieved significant word recognition rates due to complexities in Telugu language. However, the proposed segmentation algorithm offers an effective solution by accurately segmenting words into characters while preserving their distinctive features. The inception network stands out in extracting features from handwritten Telugu language, leveraging 1×1 , 3×3 , and 5×5 convolutional filters. Its integration of 1×1 convolutions ensure parameter efficiency, resulting in a more resource-efficient design compared to VGG-19. Additionally, the dictionary model adeptly addresses issues with similar and confusing characters. With a word recognition rate of 78%, approaching that of VGG-19 but with fewer parameters, this model presents a promising solution for automating handwritten Telugu text.





REFERENCES

- [1] M. Kumar, M. K. Jindal, and R. K. Sharma, "Review on OCR for handwritten indian scripts character recognition," in *Communications in Computer and Information Science*, vol. 205 CCIS, 2011, pp. 268–276.
- [2] J. He, Q. D. M. Do, A. C. Downton, and J. H. Kim, "A comparison of binarization methods for historical archive documents," in *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, 2005, vol. 2005, pp. 538–542, doi: 10.1109/ICDAR.2005.3.
- [3] C. C. Tappert, C. Y. Suen, and T. Wakahara, "The state of the art in on-line handwriting recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 8, pp. 787–808, 1990, doi: 10.1109/34.57669.
- [4] A. K. Das and B. Chanda, "A fast algorithm for skew detection of document images using morphology," *International Journal on Document Analysis and Recognition*, vol. 4, no. 2, pp. 109–114, Dec. 2001, doi: 10.1007/PL00010902.
- [5] G. Louloudis, B. Gatos, I. Pratikakis, and C. Halatsis, "Text line and word segmentation of handwritten documents," *Pattern Recognition*, vol. 42, no. 12, pp. 3169–3183, Dec. 2009, doi: 10.1016/j.patcog.2008.12.016.
- [6] J. Ryu, H. Il Koo, and N. I. Cho, "Word segmentation method for handwritten documents based on structured learning," *IEEE Signal Processing Letters*, vol. 22, no. 8, pp. 1161–1165, Aug. 2015, doi: 10.1109/LSP.2015.2389852.
- [7] P. Sahare and S. B. Dhok, "Multilingual character segmentation and recognition schemes for Indian document images," *IEEE Access*, vol. 6, pp. 10603–10617, 2018, doi: 10.1109/ACCESS.2018.2795104.
- [8] H. N. D. Bebartha and S. Mohanty, "Algorithm for segmenting script-dependant portion in a bilingual optical character recognition system," *Pattern Recognition and Image Analysis*, vol. 27, no. 3, pp. 560–568, Jul. 2017, doi: 10.1134/S1054661817030142.
- [9] H. Fujisawa, Y. Nakano, and K. Kurino, "Segmentation methods for character recognition: from segmentation to document structure analysis," *Proceedings of the IEEE*, vol. 80, no. 7, pp. 1079–1092, Jul. 1992, doi: 10.1109/5.156471.
- [10] U. Garain and B. B. Chaudhuri, "Segmentation of touching characters in printed Devnagari and Bangla scripts using fuzzy multifactorial analysis," *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, vol. 32, no. 4, pp. 449–459, Nov. 2002, doi: 10.1109/TSMCC.2002.807272.
- [11] H. R. Mamatha and K. Srikantamurthy, "Morphological operations and projection profiles based segmentation of handwritten Kannada document," *International Journal of Applied Information Systems*, vol. 4, no. 5, pp. 13–19, Oct. 2012, doi: 10.5120/ijais12-450704.
- [12] P. Inkeaw, J. Bootkrajang, P. Charoenkwan, S. Marukatat, S. Y. Ho, and J. Chaijaruwanich, "Recognition-based character segmentation for multi-level writing style," *International Journal on Document Analysis and Recognition*, vol. 21, no. 1–2, pp. 21–39, Jun. 2018, doi: 10.1007/s10032-018-0302-5.
- [13] W. Huang, Y. Qiao, and X. Tang, "Robust scene text detection with convolution neural network induced MSER trees," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 8692 LNCS, no. PART 4, 2014, pp. 497–511.
- [14] M. Mathew, M. Jain, and C. V. Jawahar, "Benchmarking scene text recognition in Devanagari, Telugu and Malayalam," in *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, Nov. 2017, vol. 7, pp. 42–46, doi: 10.1109/ICDAR.2017.364.





- [15] K. C. Prakash, Y. M. Srikar, G. Trishal, S. Mandai, and S. S. Channappayya, "Optical character recognition (OCR) for Telugu: database, algorithm and application," *Proceedings-International Conference on Image Processing, ICIP*, pp. 3963–3967, 2018, doi: 10.1109/ICIP.2018.8451438.
- [16] G. Tong, Y. Li, H. Gao, H. Chen, H. Wang, and X. Yang, "MA-CRNN: a multi-scale attention CRNN for Chinese text line recognition in natural scenes," *International Journal on Document Analysis and Recognition*, vol. 23, no. 2, pp. 103–114, Jun. 2020, doi: 10.1007/s10032-019-00348-7.
- [17] Y. Weng and C. Xia, "A new deep learning-based handwritten character recognition system on mobile computing devices," *Mobile Networks and Applications*, vol. 25, no. 2, pp. 402–411, Apr. 2020, doi: 10.1007/s11036-019-01243-5.
- [18] H. Zhao, Y. Hu, and J. Zhang, "Reading text in natural scene images via deep neural networks," in *Proceedings - 4th Asian Conference on Pattern Recognition, ACPR 2017*, Nov. 2018, pp. 49–54, doi: 10.1109/ACPR.2017.25.
- [19] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2015, doi: 10.48550/arXiv.1409.1556.
- [20] C. Szegedy *et al.*, "Going deeper with convolutions," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 07-12-June-2015, pp. 1–9, 2015, doi: 10.1109/CVPR.2015.7298594.
- [21] X. Ma, B. Li, Y. Zhang, and M. Yan, "The Canny edge detection and its improvement," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 7530 LNAI, 2012, pp. 50–58.
- [22] T. Zhang, C. Li, N. Cao, R. Ma, S. H. Zhang, and N. Ma, "Text feature extraction and classification based on convolutional neural network (CNN)," in *Communications in Computer and Information Science*, vol. 727, 2017, pp. 472–485.
- [23] M. S. Velpuru, G. Tejasree, and M. R. Kumar, "Telugu handwritten character dataset," *IEEE Dataport*, 2020, doi: 10.21227/Mw6a-D662.
- [24] N. S. Rani, S. K. Verma, and A. Joseph, "A zone based approach for classification and recognition of Telugu handwritten characters," *International Journal of Electrical and Computer Engineering*, vol. 6, no. 4, pp. 1647–1653, Aug. 2016, doi: 10.11591/ijece.v6i4.10553.
- [25] D. K. Rao and A. Negi, "Orthographic properties based telugu text recognition using hidden markov models," in *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, Nov. 2017, vol. 5, pp. 45–50, doi: 10.1109/ICDAR.2017.327.

BIOGRAPHIES OF AUTHORS







Buddaraju Revathi     is a Research Scholar at Koneru Lakshmaiah Education Foundation and has received M.Tech Degree from S R K R Engineering College. Her areas of interest are wireless communication, natural language processing, and data science. She can be contacted at email: buddaraju.revathi@gmail.com.



Dr. M.V.D. Prasad     is working as an Associate Professor within the Signal and Image Processing Research group at the Department of Electronics and Communication Engineering, Koneru Lakshmaiah Education Foundation University. Possessing a Ph.D. in Electronics and Communication Engineering, he is a highly skilled education professional. He is specializing in computer vision, image processing, natural language processing, speech technology, artificial intelligence, and data science. Leveraging extensive research experience, he has contributed to esteemed journals with his publications. He can be contacted at email: mvd_ece@kluniversity.in.



Naveen Kishore Gattim     has completed his Ph.D. from KL University. He is a faculty member at Sasi Institute of Technology and Engineering. His areas of interest are artificial intelligence and deep learning, image processing, and data science. He can be contacted at email: drnaveen@sasi.ac.in.