Optical Character Recognition using CRNN

Kambala Monica Sai, Haritha Chandrika P, Kasim Bebe, G. S Roja Pramila, G. Sankara Rao

Abstract*: Optical Character Recognition (OCR) is a computer vision technique which recognizes text present in any form of images, such as scanned documents and photos. In recent years, OCR has improved significantly in the precise recognition of text from images. Though there are many existing applications, we plan on exploring the domain of deep learning and build an optical character recognition system using deep learning architectures. In the later stage, this OCR system is developed to form a web application which provides the functionalities. The approach applied to achieve this is to implement a hybrid model containing three components namely, the Convolutional Neural Network component, the Recurrent Neural Network component and the Transcription component which decodes the output from RNN into the corresponding label sequence. The process of solving problems involving text recognition required CNN to extract feature maps from images. These sequence of feature vectors undergo sequence modeling through the RNN component predicting label distributions which are later translated using the Connectionist Temporal Classification technique in the transcription layer. The model implemented acts as the backend of the web application developed using the Flask web framework. The complete application is later containerized into an image using Docker. This helps in easy deployment on the application along with its environment across any system.

Keywords*: Artificial Intelligence, Connectionist Temporal Classification (CTC), Convolutional Neural Network (CNN), Flask web framework, Long-Short Term Memory (LSTM), Recurrent Neural Network (RNN)

I. INTRODUCTION

Optical Character Recognition (OCR) is a technological advancement into a Digital Era. Industries such as Education, Finance, Legal, Healthcare, Security and Surveillance and many more applied Optical Character Recognition to their services for more effortless processing and reduced consumption of resources. Humans can naturally recognize various objects, text and patterns.

Revised Manuscript Received on June 30, 2020.
***** Correspondence Author

Kambala Monica Sai*, Computer Science & Engineering, Gayatri Vidya Parishad College of Engineering for Women, Visakhapatnam, India. Email: monica.sai7@gmail.com

Haritha Chandrika Panuganti, Computer Science & Engineering, Gayatri Vidya Parishad College of Engineering for Women, Visakhapatnam, India. Email: harithachandrika99@gmail.com

Kasim Bebe, Computer Science & Engineering, Gayatri Vidya Parishad College of Engineering for Women, Visakhapatnam, India. Email: kasimbebe687@gmail.com

G. S. Roja Pramila, Computer Science & Engineering, Gayatri Vidya Parishad College of Engineering for Women, Visakhapatnam, India. Email: rosygangupomu@gmailc.com

G. Sankara Rao, Computer Science & Engineering, Gayatri Vidya Parishad College of Engineering for Women, Visakhapatnam, India. Email: sankararao@gvpcew.ac.in

In the case of digital devices, this pattern recognition becomes a very complicated process. Any image or scanned document is a digital file which comprises a collection of pixel values. In order to process such graphics files, the device must localize and detect the region of interest and then recognize the patterns present in the region. Such information obtained after digitalization of physical data can be used for further analysis and processing. In the early stages of development of optical character recognition systems, special fonts were required to be created which could be converted to digitally-readable text. The first font which was used for such purpose was called OCR7-B. This font type is considered as a standard font which is used by the finance industry to store and manage the sensitive data of the credit or debit cards. Later in the 1960s, OCR became a contributory factor in the digitalization of mails to assist in their sorting by the postal services. Digitizing old newspapers and scripts which contain important historical information was also another initial application of OCR. Highly industrialized countries such as the United States, Germany, Great Britain, Canada were early adopters to advance into the digital era using the optical character recognition technology.

II. MOTIVATION

In order to achieve in-house document standards efficiently, OCR gives businesses a head start to automate their workflow and fully or partially eliminate the need for paper workflow. Custom designed algorithms to implement and apply high-level OCR can enable many startups and mid-scale companies to gain profits by automating their workflow. Many businesses globally are employing OCR to digitize physical documents by capturing and processing the data. This service becomes a necessity whenever a smartphone is used for validation. One of the most common examples includes entry pass scanners at cinemas, concerts and billing counters. Similarly, this technology can also be used for identity verification at secured places like airports.

The automation of such mundane processes enhances the efficient involvement of resources in salient services. This is transparent in case of the financial industry when on a regular day, humans have to deal with millions of invoices and receipts. It can be also observed that using OCR, searching through millions of scripts and writings is automated and is also more effective than the naked eye. Every day the human brain performs many activities which enables humans to perform the activities that they normally perform. Optical character recognition is also one such activity that human intelligence provides us. Every time we look at something, our eyes process it and try to recognize it. The human brain performs computations to find patterns and analyze the input.

This is the reason we can distinguish between different letters and numbers, and generate a context. A rejuvenated technology such as OCR with applications ranging from daily activities to solving crimes is fascinating to work with when combined with the longing to learn Artificial Intelligence. Even though there are multiple algorithms and techniques existing regarding optical character recognition systems, the research in this technology is still striving to achieve better results and performance. The liveliness in the work undergoing has been an immense motivation to implement a simple optical character recognition system using deep learning architectures. This motivation resulted in a web application developed using the Flask framework which provides optical character recognition functionality enclosed to data similar to the database recommended in the future sections.

III. PROPOSED SYSTEM

Optical Character Recognition pipeline comprises various phases which help in the development of a better performing model. The major phases include text detection and text recognition. The proposed system primarily focuses on the text recognition phase. The proposed optical character recognition pipeline comprises the following phases:

A. Image Preprocessing

Image preprocessing is a crucial phase which helps maintain accuracy. Sometimes noisy and distorted images can hamper the performance of the OCR system. The input read in the form of pixel values from the HDF5 file resulting from the I/O module must be preprocessed in order to avoid conditions which might hamper the model performance. The standard preprocessing techniques applied in these modules are as follows:

▪ *Grayscale Conversion:* Generally. the captured images contain three channels in the BGR format. When images with multiple channels are used for model training, the process consumes very high storage and processing power. Each channel requires 8 bits to represent the 256 values ranging from 0 to 255. This results in a total of 256^3 possible values. Whereas, a grayscale image has a single channel so it requires to store only 256 possible values ranging from 0 to 255. So, in order to optimize this consumption, it recommended converting the image into a single format which is the grayscale version of the image.

▪ *Image Padding/Resizing:* Image padding introduces new pixels around the edges of an image. The border provides space for annotations or acts as a boundary when using advanced filtering techniques. This technique is used to assure that all the input images are of the same size. The size of (128 x 32) is applied as the input image size. So, image padding is applied to make the image compatible with the required dimensions.

▪ *Dimensionality Expansion:* Dimensionality expansion is a mathematical technique to increase the multispectral data dimensionality in a non-linear fashion. The input dimension required by the neural network architecture is (128 x 32 x 1). So, in order to make the images compatible with the architecture's input, the dimensionality of the image is expanded.

▪ *Normalization:* In image processing, normalization is a process that alters the range of pixel intensity values. Normalization is also called contrast stretching or histogram stretching. Normalization transforms an n-dimensional grayscale image with intensity values in the range (Min, Max), into a new image with pixel intensity values in the range (newMin, newMax).

▪ *Output Label Encoding:* Initially, all the output labels of images are in the form of strings. But since neural networks cannot process string characters it is required to encode these characters. Each character is encoded in a numerical value by processing it through a user-defined encoding function. This function converts each character into an integer corresponding to the character's position in the Alphabet. In order to maintain equal size among all the encoded label sequences to make them compatible with the output shape of the RNN component in the architecture, the padding technique from TensorFlow is applied.

B. Text Recognition

The implemented neural network model is a Convolutional Recurrent Neural Network (CRNN) which is a combination of Deep Convolutional Neural Network (Deep-CNN) and Recurrent Neural Network (RNN). The architecture of the convolutional layers is based on the VGG-VeryDeep architectures. Apart from the deep convolutional layers, the architecture also contains recurrent layers. In order to train such a deep neural network model a technique called Batch Normalization is applied. Two batch normalization layers are used to accelerate the training process. CRNN architecture contains the convolutional layers which extract feature maps automatically. This is followed by a recurrent network used for predicting the text from each frame of the feature map sequence. This per-frame prediction by the recurrent network is translated into a label sequence using a transcription layer. The transcription layer used Connectionist Temporal Classification (CTC) loss function to predict the output at each epoch.

IV. ARCHITECTURE

The text recognition phase of the optical character recognition system uses a CRNN architecture which is a combination of CNN and RNN along with transcription layers and Connectionist Temporal Classification (CTC).

This neural network architecture integrates the functionalities of feature extraction, sequence modeling and transcription into a consolidated framework. The architecture of the model can be depicted as follows:

/

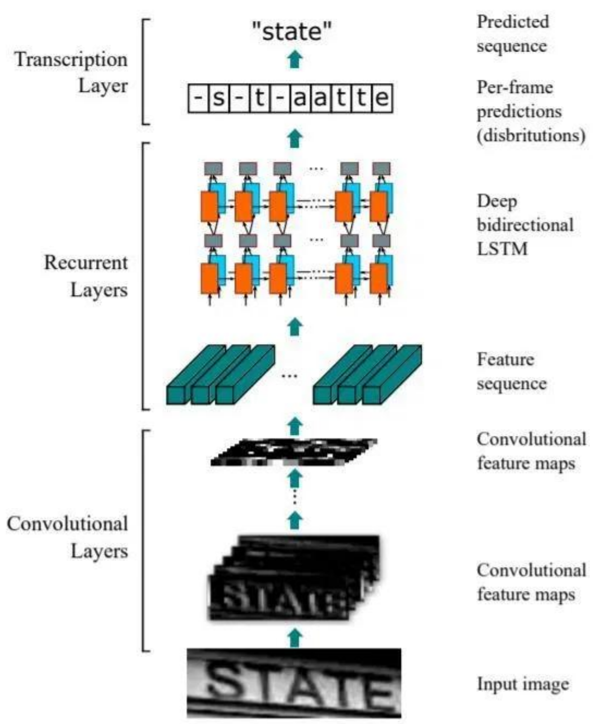


Fig. 1: CRNN Architecture

The CRNN architecture does not require character segmentation. The convolution part of the neural network extracts features maps from the input region which contains the text detected regions. The deep bidirectional recurrent neural network following the CNN predicts the encoded label sequence which when decoded results in the final output. This translation is performed by the transcription layer. The transcription layer uses the CTC loss to convert the per-frame prediction into a label sequence.

The following is the brief architecture flow which is later explained in detail:

1. The preprocessed image given as input to the CRNN architecture.
2. This input image is passed on to the convolution component for feature extraction.
3. The convolutional feature maps generated are used to form receptive fields resulting in a sequence of feature vectors.
4. The output from the CNN component is passed as input to the RNN component for sequence modeling.
5. The layers in RNN predict label distribution for every frame in the feature sequence.
6. The per-frame label distributions generated from the bidirectional LSTM layers are passed through the transcription component.
7. These label distributions are converted into label sequences by applying the Connectionist Temporal Classification (CTC).
8. The label sequences obtained are then converted into the format of required label using the CTC decoder.

A. Feature Extraction

The first phase in the CRNN model architecture is the feature extraction phase. The component is a combination of convolutional layers which are constructed by combining convolutional and max-pooling layers from a standard CNN model excluding the full connected layers. A sequence of feature vectors is generated by the convolutional layers. Each feature vector, in particular, is generated from left to right on the feature maps by the column. The width of each column is fixed to a single pixel. This means that each i-th feature vector is the sequence of the i-th columns of all the feature maps. A standard CNN contains a combination of convolution layers, max-pooling layers and element-wise activation function to operate on the local regions. Each column present in the feature map generated corresponds to the associated rectangular block of pixels on the original image. This rectangular region is also called a receptive field and each of it considered as the image descriptor of that region. After the feature maps are obtained, a sequence of feature vectors is extracted.

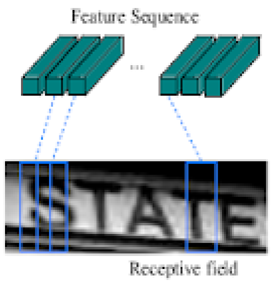


Fig 2: Receptive Field

B. Sequence Modeling

The convolution component of the architecture is followed by a deep bidirectional Recurrent Neural Network as the recurrent layer component. The recurrent layers in the component predict the label distribution for each frame in the feature sequence obtained from the convolution component. The primary advantage of using recurrent neural networks is its rich capability of capturing contextual information from the sequence. Such contextual information obtained from the feature sequence allows recognition of sequences in images more stable rather than considering each character or frame independently.

The recurrent layers predict a label distribution for every frame in the feature sequence depicted above. In general, a conventional RNN unit contains a self-connected hidden layer between its input and output layers. It updates its internal state after every time a frame is received from the sequence during an epoch. The renewed state is generated from a nonlinear function that takes both the current input and past state as its input parameters. In this way, historical contexts are captured and utilized for predictions by the recurrent layers. A traditional RNN however, is troubled with the vanishing gradient problem, which limits the range of context it can store and adds burden to the training process. Long-Short Term Memory or LSTM is a kind of RNN unit that is designed in particular to handle this problem. An LSTM unit consists of a memory cell along with three multiplication dates, namely the input gate, output gate and the forget gate.

Theoretically, the memory cell stores the previous contexts which can be stored for a long period of time because of the input and output gates.

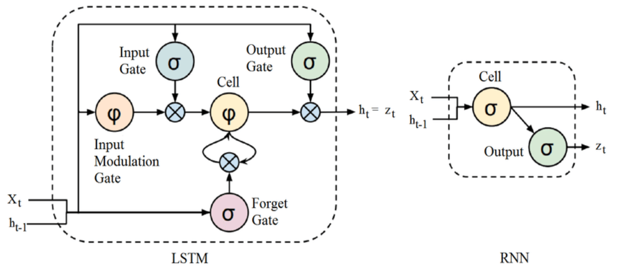


Fig 3: Basic LSTM cell with RNN

The contexts data present in the memory cell can be cleared by the forget gate. This specialized design of LSTM allows the capturing of long-range dependencies which are commonly encountered in image-based sequences. A traditional LSTM is directional and uses only past contexts. But, while applying LSTMs for image-based sequences, contexts from both the directions are required. Therefore, to achieve this, a combination of two LSTMs are used. One LSTM is forward and the other is backward, which results in a bidirectional LSTM. In order to form a deep bidirectional LSTM, multiple bidirectional LSTMs are stacked upon each other. A deep architecture is preferred than a shallow one as it helps in achieving improved performance. In order to bridge the convolutional layers and the recurrent layers in the architecture, a custom network layer called "Map-to-Sequence" is designed.

C. Transcription

The transcription layers are used for converting the per-frame label distribution obtained from RNN into a label sequence. Mathematically, this is carried out by finding the maximum probability distribution on the pre-frame predictions. A Connectionist Temporal Classification (CTC) acts as the Transcription Layer in the CRNN architecture. Generally, while applying supervised learning algorithms on problems dealing with sequence generation there exists a problem which requires labeling the input image at each and every time step. Consider an image of 128 pixels, when applying RNN it takes 128-time steps to pass as input. In that case, it is required to label each time step in the training dataset. In order to handle this problem, CTC can be applied as it takes care of the input labeling at each time step and requires to input only the label sequence of the original input image. Another important functionality of CTC is that in case of images with text containing multiple occurrences of the same character, there is a high possibility of ambiguity regarding the count of the character occurrences. Consider an image containing the word "boot". Due to the image dimensions and the receptive fields, the RNN predicts the output as ['b', 'b', 'o', 'o', 'o', 't', 't']. Traditionally, to predict the text output finally, it required to merge the consecutively repeating characters. But in such cases, the output will be predicted as "bot" instead of "boot". CTC handles this ambiguity by inserting a blank character in between the continuously repeated characters. In this case, it is generated as ['b','b','o','o,' '-', 'o', 't', 't']. This blank character is later excluded while decoding text for predicting the output.

Every neural network architecture requires a loss function in order to achieve optimized weights. CTC loss is a different kind of loss function which computes the loss by taking the negative logarithm of the summation of all possible alignments of the text present in the image. Even though this is an expensive computation process, CTC uses dynamic programming to perform this operation making it faster. After the predictions from the RNN component, the output text must be decoded. This is performed by the CTC by considering the characters with the highest probability at each time step and eliminating the duplicate and blank characters. CTC loss can also be applied to problems dealing with speech recognition.

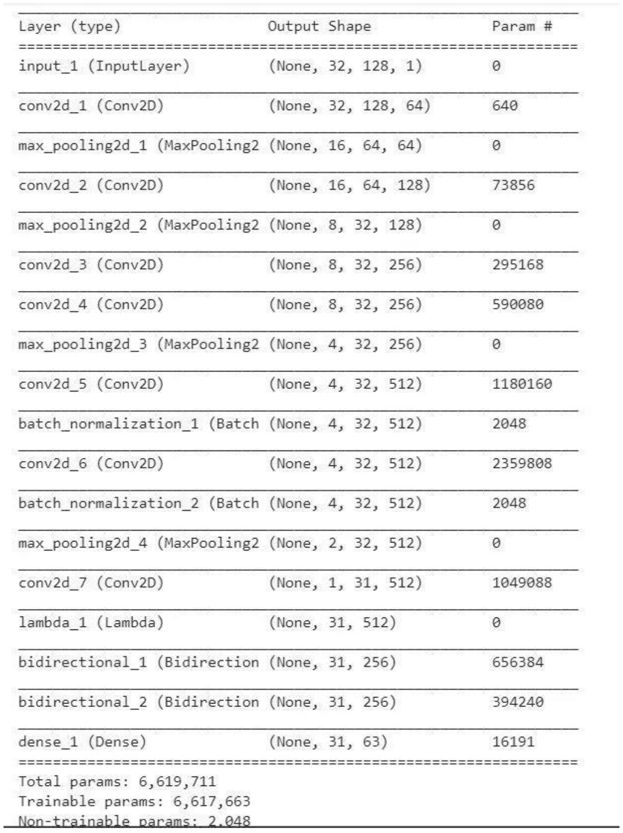

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         (None, 32, 128, 1)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 32, 128, 64)       640
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 16, 64, 64)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 16, 64, 128)       73856
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 8, 32, 128)        0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 8, 32, 256)        295168
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 8, 32, 256)        590080
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 4, 32, 256)        0
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 4, 32, 512)        1180160
_________________________________________________________________
batch_normalization_1 (Batch (None, 4, 32, 512)        2048
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 4, 32, 512)        2359808
_________________________________________________________________
batch_normalization_2 (Batch (None, 4, 32, 512)        2048
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 2, 32, 512)        0
_________________________________________________________________
conv2d_7 (Conv2D)            (None, 1, 31, 512)        1049088
_________________________________________________________________
lambda_1 (Lambda)            (None, 31, 512)           0
_________________________________________________________________
bidirectional_1 (Bidirection (None, 31, 256)           656384
_________________________________________________________________
bidirectional_2 (Bidirection (None, 31, 256)           394240
_________________________________________________________________
dense_1 (Dense)              (None, 31, 63)            16191
=================================================================
Total params: 6,619,711
Trainable params: 6,617,663
Non-trainable params: 2,048
```

Fig. 4: CRNN Layers

The layers present in the CRNN architecture can be described as follows:

1. The input dimensions of the architecture are defined as (128 x 32) where the image height is 32 pixels and width is 128 pixels with a single channel.
2. The CNN component of the architecture uses 7 convolution layers in which 6 layers have the kernel size of (3, 3) and the final one of size (2, 2).
3. After each and every layer, the number of filters to perform the convolution operation is increased from 64 to 512 incrementally.
4. The CNN component also contains 4 max-pooling layers where 2 layers are of size (2, 2) and others with size (2, 1).

The pooling layers are used to extract features from the output of the previous convolution layer (larger image).

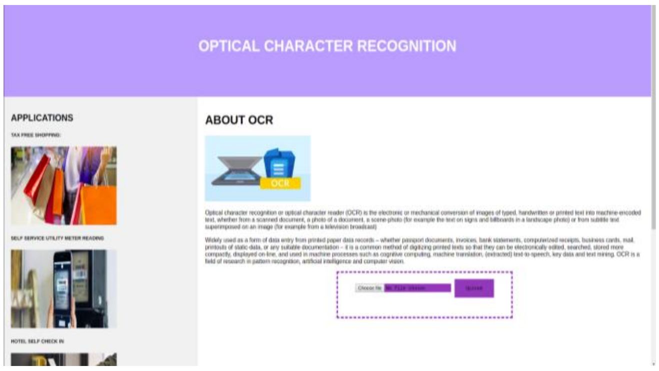
5. The output of the pooling layers is a reduced image which contains the feature maps of the larger image.
6. Since the architecture is applied on unstructured data, batch normalization is a technique used for improving the speed, performance and stability of the neural network architecture. It normalizes the layer by applying the re-centering and re-scaling methods.
7. In this architecture, two batch normalization layers are added after the fifth and sixth convolution layers.
8. This is followed by the bidirectional LSTM layers each 128 units. The output generated from each of the RNN layers is of size (batch_size, 31, 63) where 63 is the number of output classes including the blank character (lower-case alphabet + upper-case alphabet + digits).
9. After the model predicts the probability of each class at every time step, the transcription function which is the CTC decoder is applied to convert it into actual input.

V. RESULTS

The web application which offers the functionality of text recognition from the model trained is developed using the Flask framework. Flask is a lightweight web framework which provides tools, libraries and technologies to build a web application. The saved model from the previous model is loaded in the web application and applied to perform the text recognition task on the image uploaded from the front-end.

The environment of such applications which use deep learning is very important to successfully use its functionalities. So, in order to avoid dependencies related issues, an image of the application including its complete environment is created using Docker. This allows for easy deployment and usage of the application across multiple devices.

The web application screens are given as follows:



Optical character recognition or optical character reader (OCR) is the electronic or mechanical conversion of images of typed, handwritten or printed text into machine-encoded text, whether from a scanned document, a photo of a document, a scene-photo (for example the text on signs and billboards in a landscape photo) or from subtitle text superimposed on an image (for example from a television broadcast).

Widely used as a form of data entry from printed paper data records – whether passport documents, invoices, bank statements, computerized receipts, business cards, mail, printouts of static data, or any suitable documentation – it is a common method of digitizing printed texts so that they can be electronically edited, searched, stored more compactly, displayed on-line, and used in machine processes such as cognitive computing, machine translation, (extracted) text-to-speech, key data and text mining. OCR is a field of research in pattern recognition, artificial intelligence and computer vision.

Fig. 5: Homepage

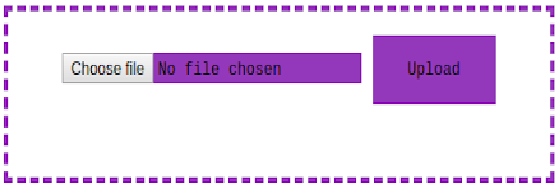


Fig. 6: File Upload

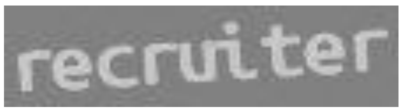
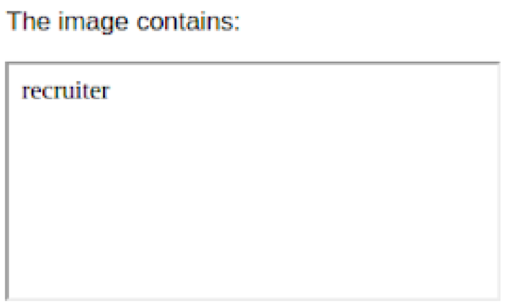


Fig. 7: Sample Input Image



The image contains:

recruiter

Fig. 8: Recognized Output

VI. CONCLUSION

The objective of the project to develop a web application which simulates an optical character recognition system by providing the functionality of text recognition is satisfied. This functionality is successfully implemented using deep learning by applying a CRNN architecture which is a combination of CNN and RNN. The development of this project using this architecture allowed the exploration of different neural network architectures leading to a hybrid structure of using CNNs and RNNs along with CTC loss. The dataset used for training this architecture is the "MJSynth" synthetic word dataset by the University of Oxford which contains 9 million images of 90k unique English words. The test accuracy when considering 10k images with condition that every character to be exactly correct is obtained to be 59.9%. The scope of this project can be further enhanced by extending it with certain advancements such as applying state-of-the-art text detection techniques. If there is a requirement to deploy the application for real-time usage, it should give state-of-the-art results. Hence in order to achieve good results application-wise, tools such as Google Tesseract OCR engine, Google Cloud Vision API, AWS Textract can be used. In the case of research-oriented work, existing pre-trained models such as the VGG architecture, ResNet can be explored, analyzed and enhanced along with efficient text detection techniques. These advanced approaches can enable the usage of the OCR system on various types of inputs such as scanned documents, real-scene images, vehicle number plates and even QR code and barcodes. Artificial Intelligence is a field which is currently witnessing large-scale research and Optical Character Recognition is one of the applications experiencing high demand in terms of performance. Hence continuous development and enhancement both application-wise and research-wise can be carried out on this project.

REFERENCES

1. "A Review on the Various Techniques used for Optical Character Recognition", Pranob K Charles, V. Harish, M. Swathi, Ch. Deepthi, International Journal of Engineering Research and Applications, Vol. 2, Issue 1, pp. 659-662, Jan-Feb 2012.

2. "Improving Optical Character Recognition Techniques", Nitin Ramesh, Aksha Srivastava, K. Deeba, International Journal of Engineering & Technology.
3. "A State-of-the-Art Survey on Deep Learning Theory and Architectures", Md Zahangir Alom, Tarek M. Taha, Chris Yakopcic, Stefan Westberg, Paheding Sidike, Mst Shamima Nasrin, Mahmudul Hasan, Brian C. Van Essen, Abdul A. S. Awwal and Vijayan K. Asari. www.mdpi.com/journal/electronics
4. "An Overview of the Tesseract OCR Engine", Ray Smith, Google Inc.
5. "An End-to-End Trainable Neural Network for Image-based Sequence Recognition and its Application to Scene Text Recognition", Baoguang Shi, Xiang Bai, Cong Yao, arXiv.
6. "Optical Character Recognition Technique Algorithms", N. Venkata Rao, Dr. A.S.C. Sastry, A.S.N Chakravarthy, Kalyanchakravarthi P, Journal of Theoretical and Applied Information Technology, 20th January 2016, Vol. 83, No. 2, ISSN: 1992-8645.

AUTHORS PROFILE

Kambala Monica Sai, final year Computer Science and Engineering undergraduate from Gayatri Vidya Parishad College of Engineering for Women, Visakhapatnam. A Google's TalentSprint Women Engineers 2019 Scholar and Research Intern in Biometrics at International Institute of Information Technology Hyderabad. Earned Microsoft Technology Associate Certification in "Introduction to Programming Using Python". A Bertelsmann Technology Scholar on Udacity for Deep Learning Nanodegree and Intel Edge AI Technology Scholar on Udacity for Intel Edge AI for IoT Developers Nanodegree. Gained experience in computer vision concepts through work on face recognition using holistic techniques such as PCA and Improved PCA, feature-based technique using face landmarks extraction and deep learning using Convolution Neural Networks.

Haritha Chandrika. P, final year Computer Science and Engineering undergraduate at Gayatri Vidya Parishad College of Engineering for Women, Visakhapatnam. Experience in projects on Credit Card fraud detection, Breast Cancer Prediction using PCA, Text Summarization implemented using Data Mining, Plant Disease Detection.

Kasim Bebe, final year Computer Science and Engineering undergraduate at Gayatri Vidya Parishad College of Engineering for Women, Visakhapatnam. Experience in project on Breast Cancer Prediction using Naive Bayes Algorithm. The keen interest built through this project in Machine Learning is an inspiration that led to the work with Deep Learning techniques.

G.S.Roja Pramila, final year Computer Science and Engineering undergraduate at Gayatri Vidya Parishad College of Engineering for Women, Visakhapatnam. Experience in project on Health care Prediction using Naive Bayes Algorithm.

G. Sankara Rao is currently working as an Assistant Professor in Department of Computer Science & Engineering at Gayatri Vidhya Parishad College of Engineering for Women, Visakhapatnam. He has more than 12 years of experience in Teaching field. His research interests include Deep learning, Wireless Sensor Networks, Networking and Security. He completed M.Tech from Andhra University, and pursuing Ph.D at JNTUK University.