A scene perception system for visually impaired based on object detection and classification using CNN

Lalita Moharkar[1,*] , *Sudhanshu* Varun [2], *Apurva* Patil [3] , and *Abhishek* Pal [4]

[1]Assistant professor, Xavier Institute of Engineering, Mumbai, India
[2] Student of Electronics and Telecommunication Department, Xavier Institute of Engineering, Mumbai, India
[3] Student of Electronics and Telecommunication Department, Xavier Institute of Engineering, Mumbai, India
[4] Student of Electronics and Telecommunication Department, Xavier Institute of Engineering, Mumbai, India

Abstract. In this paper we have developed a system for visually impaired people using OCR and machine learning. Optical Character Recognition is an automated data entry tool. To convert handwritten, typed or printed text into data that can be edited on a computer, OCR software is used. The paper documents are scanned on simple systems with an image scanner. Then, the OCR program looks at the image and compares letter shapes to stored letter images. OCR in English has evolved over the course of half a century to a point that we have established application that can seamlessly recognize English text. This may not be the case for Indian languages, as they are much more complex in structure and computation compared to English. Therefore, creating an OCR that can execute Indian languages as suitably as it does for English becomes a must. Devanagari is one of the Indian languages spoken by more than 70% of people in Maharashtra, so some attention should be given to studying ancient scripts and literature. The main goal is to develop a Devanagari character recognition system that can be implemented in the Devanagari script to recognize different characters, as well as some words.

1. Introduction

In recent years, handwriting recognition has been one of the fascinating and demanding fields of research in the field of image processing and pattern recognition. It significantly contributes to the development of an automation method and in various applications, it can enhance the interaction between humans and machine. Several research works focused on new technologies and methods that would reduce the processing time while providing greater accuracy in recognition. Handwritten Indian script (Devanagari) is a tough task due to the different characteristics of these scripts, such as their broad character collection, complex form, presence of modifiers, presence of compound characters and character similarity. Devanagari script is mostly useful for the purpose of writing and recording most Indian languages including Hindi, Marathi, Sindhi, Nepali, Sanskrit, Konkani, Maithili, etc. One of Indian script's defining features is its variety of sounds that it must support. As there's usually a letter in Indian languages for each of the phonemes, the collection of alphabets appears to be very large. Devanagari developed through various transformations from the ancient Brahmi script. India's official Hindi national language, written in the Devanagari alphabet, has more than 500 million speakers. The subsequent characteristics will regulate the complex nature of the Devanagari script. The main idea is to train a model with thousands of images containing Devanagari scripts in it and make computer understands every letter and word by running the image over and over through the machine learning algorithms which will then be used to recognize when and when these same words or letters can be handwritten or printed on a machine. So making it much faster and more efficient as the number of such documents increases over the time. This can also be trained to recognize and study antique scripts and literature. We will be introducing an offline method for handwritten recognition of Devanagari characters in this project. Offline character recognition is a method where the letters / words to be recognized are in the form of scanned documents that are stored in the greyscale format. Later this scanned data is passed through certain stages that are essential for the identification of superior characters. Postprocessing steps to prepare images.

2. Major Steps

These steps have been successfully implemented. Once these steps have been successfully implemented on every image present in the image data set next we need to develop algorithms that's can be used to train and test the model and later predict how well the model has been trained and then we install the library for voice output.
Handwriting character Recognition Technology has been refining with the help of pattern recognition and image processing for a few decades. Various software computing methods that are used in other types of pattern and image recognition can as well be used for Devanagari character recognition.
These are the following Steps:

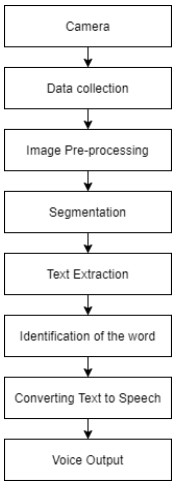

Fig. 1. Steps wise representation

2.1 Camera

The Pi camera module is a light weight, compact camera. It communicates with Pi via the serial interface protocol for the MIPI camera. It is usually used in image processing, in machine learning or in project surveillance. It is widely used in surveillance drones since camera payload is very small. Among these modules Pi can also use regular USB webcams that are used in conjunction with computers. Pi camera is used to take the image and preprocessing is done using python.

2.2 Data Collection

we have generated and stored the database by making 10 different people to write few selected words and then we scanned those words and cropped the images and labeled the images of handwritten data

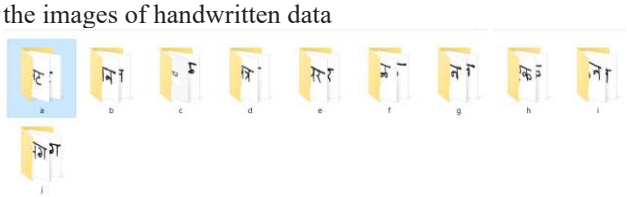

Fig. 2 . Folders of Dataset of Alphabet-wise words

Fig. 3 . Handwritten Dataset of one person

2.3 Image Pre-processing

Pre-processing consists of all the operation that needs to be done on a scanned image to improve its efficiency, so that when the actual processing is finished, less noise is present in the image.

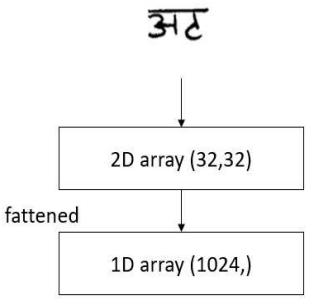

Fig. 4. Image Pre-processing

2.4 Segmentation

A sequence image of characters in the segmentation stage is decomposed into sub-images of individual character. In the proposed system, the pre-processed input image is segmented into isolated characters using a marking process to assign a number to each character. This labeling provides information on the number of characters that appear in the illustration. For the classification and reconnaissance level, each individual character is uniformly resized into 90X60 pixels.

2.5 Text Extraction

At this point all the critical features necessary for good character recognition is extracted. If the extraction of the function takes place more precisely we get better output so this is an important stage.

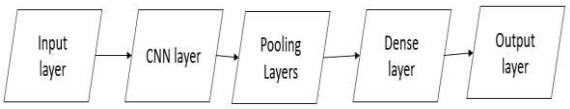

Fig. 5. Text Extraction

2.6 Identification Of Words

This stage is used for the recognition decision making segment taking advantage of the features previously extracted. It uses feed forward convolutional neural networks with several hidden layers. Such as the thick layers and the peak pooling.

2.7 Converting Text to speech

Pyttsx is a speech library cross-platform text that is independent of a platform. The key benefit of using this

library for the conversion from text to speech is that it operates offline. Pyttsx however only supports Python 2.x. Therefore, we will see pyttsx3 that is modified to work with the same code on both Python 2.x and Python 3.x. The installation command is **pip install pyttsx3**

3. Literature review

According to [1] The paper describes an off-line handwritten alphabetical character recognition system which uses multilayer feed forward neural network. To extract the features of the handwritten alphabets, a new method, called, diagonal-based extraction function is implemented. Fifty data sets are used for training the neural network, each containing 26 alphabets written by different people, and 570 different handwritten alphabetic characters are used for testing. The proposed recognition system performs quite well yielding higher levels of accuracy in recognition compared to the systems using traditional horizontal and vertical extraction methods. This system will be suitable for conversion into structural text form of handwritten documents and recognition of handwritten names. Two approaches chosen to build the reconnaissance system for the Neural Network are trained using the methods of horizontal and vertical extraction of features. A simple, off-line English alphabet character recognition system using a new type of extraction feature, namely extraction of the diagonal feature, is proposed. Two approaches are chosen to build the Neural Network recognition system using 54 features and 69 features. The neural network recognition system is trained using the horizontal and vertical extraction methods to compare the recognition efficiency of the proposed diagonal method of extraction of features. There are six different networks of recognition built up. According to [2] The failures in identifying written Devanagari characters are primarily due to incorrect touching or damaged character segmentation. Because of the upper and lower Devanagari text modifiers, several portions of two consecutive lines can also overlap. According to [3] Handwritten character recognition is currently attracting researchers ' attention due to possible applications in assisting users with blind and visually impaired technology, interaction between human and robot, automatic data entry for business documents, etc. In this work , we propose a technique for recognizing D evanagari handwritten characters using deep convolutio nary neural networks (DCNN), one of the deep-learning community's recent techniques.According to [4] Handwritten Devanagari ancient manuscript recognition system was introduced using the extraction techniques of statistical features. Intersection points, open endpoints, centroid, horizontal peak extent, and vertical peak extent characteristics are extracted in the extraction process of feature. In this work, the Convolutional Neural Network, Linear Network, Multilayer Perceptron, RBF-SVM, and random forest techniques are considered for classification. Various techniques of extraction and classification of features are considered and compared with the recognition of basic characters segmented from ancient manuscripts in Devanagari.4 features are extracted for the recognition of those characters. These characteristics are intersection and open endpoints, centroid, horizontal peak, and vertical peak range. Use all possible combinations of those features. Five classifiers were used by the authors, namely MLP, Neural Network, CNN, RBF-SVM, and Random Forest. According to [5] In this thesis, we analyse the problems and challenges for Indic scripts with automatic handwritten word recognition.We addressed the latest Unicode encoding standard and why we selected a single Unicode character as our basic recognition unit. Finally, we spoke about the problems that make HWR more difficult as opposed to Latin for Indic scripts (more characters, less real data).

4. Why is CNN important?

Neural Networks were the greatest invention of the human mind by creating a network which possesses qualities similar to those of the human brain and allows for thinking machines. Machines think, learn, and perform tasks that only with the help of these neural networks are possible for the one human. With the aid of Convolutional Neural Networks, the form of a neural network that has enabled machines to visualize images and distinguish one from another has become possible. Convolutional neural networks allow machines to perceive the world as human beings and thus become an essential principle for learning when working in computer vision. Computer vision has made computers perform several of those impossible tasks that were once managed and performed only by humans. Computer vision used in surveillance, medical imaging system, navigation, inspection has enabled computers to be smart enough to detect what's going on in the world, just like human eyes. Now computers too have eyes like humans and with the help of convolutional neural networks this is all possible. Convolutional Neural Networks are integrated into machines inspired by the neurons in human brains to provide eyes to machines and allow them for visual imaging. With the aid of Convolutional Neural Networks embedded in the system, visual imagery, video recognition, object detection- all have become possible.

5. Flow chart

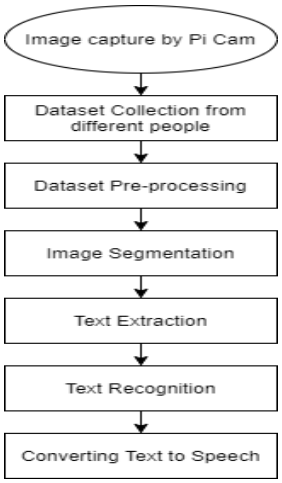

Fig. 6. Flowchart

6. Proposed Method

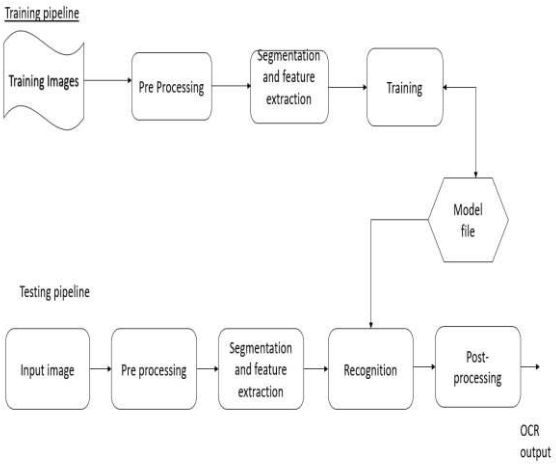

Fig. 7. Block Diagram

6.1 Convolutional Neural Networks

The Convolutional Neural Network (CNN or ConvNet) is a biologically based machine-leaning architecture that can learn from experiences such as traditional multilayer neural networks. ConvNets consists of several layers of overlapped tiling arrays of small neurons such that the original image is best represented. ConvNets is commonly used for identification of photographs and videos. Three key types of layers are used to create a ConvNet architecture .

6.1.1 Convolution Layer:

Convolution is the first layer where features are extracted from an input image. Convolution preserves the relation between pixels by using small squares of input data to learn image features. It is a mathematical operation that involves two inputs such as a matrix of images and a filter or kernel.

6.1.2 Max-Pooling Layer:

Section of pooling layers will lower the number of parameters when the images are too large. Spatial pooling also known as subsampling or down sampling which reduces each map's dimensionality but retains important information.

6.1.3 Fully-Connected Layer:

Fully connected layers are an essential component of the Convolutional Neural Networks (CNNs), which have proven to be very effective in the identification and classification of computer vision images. The CNN process begins with convolution and pooling, the image is broken down into features and analyzed separately. The product of this process feeds into a fully connected neural network structure which drives the final decision on the classification. This article provides a thorough review of CNNs, how their architecture works and how it relates to computer vision deep learning applications in real world.

7. Results

```
...: model.save("DevaModel1.h5")
Using TensorFlow backend.
Found 78200 images belonging to 46 classes.
Found 13800 images belonging to 46 classes.
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 30, 30, 32)        320
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 15, 15, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 13, 13, 64)        18496
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 7, 7, 64)          0
_________________________________________________________________
dropout_1 (Dropout)          (None, 7, 7, 64)          0
_________________________________________________________________
flatten_1 (Flatten)          (None, 3136)              0
_________________________________________________________________
dense_1 (Dense)              (None, 128)               401536
_________________________________________________________________
dense_2 (Dense)              (None, 64)                8256
_________________________________________________________________
dense_3 (Dense)              (None, 46)                2990
=================================================================
Total params: 431,598
Trainable params: 431,598
Non-trainable params: 0
```

Fig. 8. CNN Architecture Implementation

```
Epoch 45/50
31/31 [==============================] - 1s 40ms/step - loss: 0.4474 -
accuracy: 0.8724 - val_loss: 0.3232 - val_accuracy: 0.9449
Epoch 46/50
31/31 [==============================] - 1s 38ms/step - loss: 0.4412 -
accuracy: 0.8531 - val_loss: 0.1570 - val_accuracy: 0.9357
Epoch 47/50
31/31 [==============================] - 1s 40ms/step - loss: 0.4528 -
accuracy: 0.8612 - val_loss: 0.3362 - val_accuracy: 0.9469
Epoch 48/50
31/31 [==============================] - 1s 40ms/step - loss: 0.3818 -
accuracy: 0.8796 - val_loss: 0.1314 - val_accuracy: 0.9622
Epoch 49/50
31/31 [==============================] - 1s 42ms/step - loss: 0.4044 -
accuracy: 0.8765 - val_loss: 0.1751 - val_accuracy: 0.9469
Epoch 50/50
31/31 [==============================] - 1s 44ms/step - loss: 0.3647 -
accuracy: 0.8827 - val_loss: 0.2465 - val_accuracy: 0.9551
```

Fig. 9. Training the Model

We have used a Convolutional Neural Network (CNN) model in this approach. Thus, we train it using the above-mentioned pre-processed dataset. Fig.8. represents the implementation of the architecture of the Convolutional Neural Network (CNN). The layers are present as mentioned in Fig.8. The model is trained as observed from Fig.9.

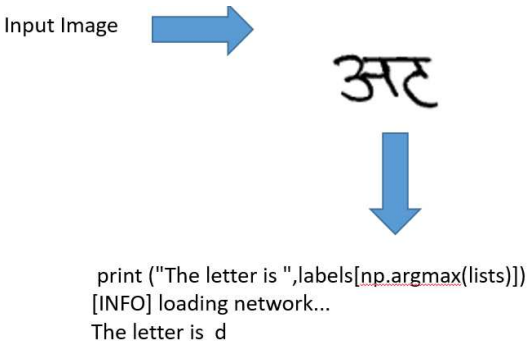

Fig. 10. Output of the Trained Model

Fig. 10. represents the output of the model after training. After the input image is provided, the model gives the respective output.

8. Conclusion

The images can be captured from the Pi cam and we can use it through the Python code for testing in the raspberry pi and then, after the text of the image is identified, we get the output in the form of sound. We proposed a CNN based method for devanagari text recognition and later convert it into speech. The accuracy obtained to recognize the text image for the dataset prepared is 94-95% using CNN as shown in Fig.10. The proposed CNN effectively learns quality related features and achieved the state-of-the-art performance.

9. Future Work

The dataset can be modified to obtain better results. The presented architecture of the Convolutional Neural Network (CNN) is optimum. But we would also try modifying the layers of the Convolutional Neural Network (CNN) which may lead to a better result. The accuracy of the model can be further increased. This can be implemented using Raspberry Pi and can be converted into a complete application which can help the visually impaired people.

10. References

[1] J. Pradeep , E. Srinivasan and S. Himavathi , IJCSIT, "*Diagonal Based Feature Extraction for Handwritten Alphabets Recognition System Using Neural Network"* , **Vol 3, No 1**, (2011)

[2] A. S. Ramteke, M. E. Rane , IJSER, *"A Survey on Offline Recognition of Handwritten Devanagari Script*" **Volume 3** (2012)

[3] M. Jangid and S. Srivastava, J.IMAGE "*Handwritten Devanagari Character Recognition Using Layer-Wise Training of Deep Convolutional Neural Networks and Adaptive Gradient Methods"* (2018)

[4] S. Narang, M K. Jindal and M. Kumar, IAC , "*Devanagari ancient documents recognition using statistical feature extraction techniques"* (2019)

[5] K. Dutta, IIITH, *"Handwritten Word Recognition for Indic & Latin scripts using Deep CNN-RNN Hybrid Networks"* (2019)