Optical Character Recognition (OCR) System for Devanagari Script

The Optical Character Recognition (OCR) system outlined in the provided code is designed to recognize and classify characters in the Devanagari script. The system follows a series of steps to preprocess, segment, localize, and classify characters within an input image. Here's an overview of each key component:

1. **\*\*Preprocessing\*\***:

- The input image is preprocessed using Gaussian blur and thresholding to enhance the contrast between characters and the background.

2. **\*\*Segmentation\*\***:

- The preprocessed image is segmented into individual characters using horizontal and vertical segmentation techniques. This step divides the image into smaller segments containing single characters.

3. **\*\*Localization\*\***:

- The segmented characters are localized on the original image to highlight their positions. This step helps visualize the boundaries of each character within the image.

4. **\*\*Classification\*\***:

- Each localized character is classified using a trained Convolutional Neural Network (CNN) model. The model predicts the character represented by each segment based on its features.

5. **\*\*Results\*\***:

- The system outputs the classified characters along with their positions in the original image. It also provides the accuracy of the classification algorithm.

6. **\*\*Technology Stack\*\***:

- The system is implemented in Python using libraries such as NumPy, OpenCV, Matplotlib, and TensorFlow for image processing, segmentation, and classification.

7. **\*\*Purpose\*\***:

- The OCR system aims to automate the process of recognizing Devanagari characters in images, which can be used for various applications such as text extraction, translation, and document digitization.

Overall, the OCR system provides a foundation for recognizing Devanagari characters in images and can be further enhanced and integrated into applications that require automated text recognition and processing.

**TECHNOLOGY USED:-**

The OCR system described in the provided code is implemented using the following technologies:

* **Python:** The core programming language used for the implementation of the OCR system.
* **NumPy:** A library for numerical computations in Python, used for efficient handling of arrays and matrices, which is essential for image processing tasks.
* **OpenCV (Open Source Computer Vision Library):** A library of programming functions mainly aimed at real-time computer vision, used for image processing tasks such as loading, preprocessing, and segmenting images.
* **Matplotlib:** A plotting library for creating static, animated, and interactive visualizations in Python, used for displaying images and visualizing the results of image processing.
* **TensorFlow:** An open-source deep learning framework developed by Google, used for building and training the Convolutional Neural Network (CNN) model for character classification.
* **Datasets:** Both hand picked data as well as by creating own datasets are used in this project.

These technologies are widely used in the field of image processing and machine learning, making them suitable choices for developing an OCR system for character recognition.

**ARCHITECTURE:-**

The architecture used in the provided OCR system includes several components working together to recognize and classify Devanagari characters in images. Here's an overview of the architecture:

* **Image Preprocessing:** The input image undergoes preprocessing steps such as Gaussian blur and thresholding to enhance the contrast and prepare it for segmentation.
* Segmentation: The preprocessed image is segmented into individual characters using horizontal and vertical segmentation techniques. This step helps isolate each character for classification.
* **Localization:** The segmented characters are localized on the original image to visualize their positions and boundaries, aiding in the identification of characters.
* Character Classification: Each localized character is classified using a trained **Convolutional Neural Network (CNN) model**. The model predicts the character represented by each segment based on its features.
* **Output Visualization:** The system provides visual outputs, such as displaying the segmented characters and their positions in the original image, along with the accuracy of the classification algorithm.
* **Technology Stack:** The architecture leverages Python as the core programming language, along with libraries such as NumPy, OpenCV, Matplotlib, and TensorFlow for image processing, segmentation, and classification tasks.

Overall, the architecture combines image processing techniques with machine learning to automate the recognition of Devanagari characters in images, demonstrating a comprehensive approach to optical character recognition.

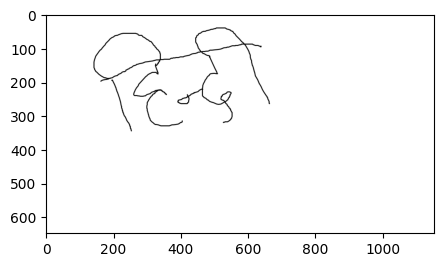
**WORKING:-**

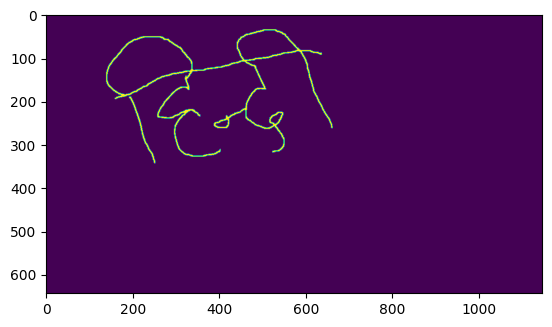
The provided OCR (Optical Character Recognition) system works by following a series of steps to recognize and classify Devanagari characters in an input image. Here's a high-level overview of how it works:

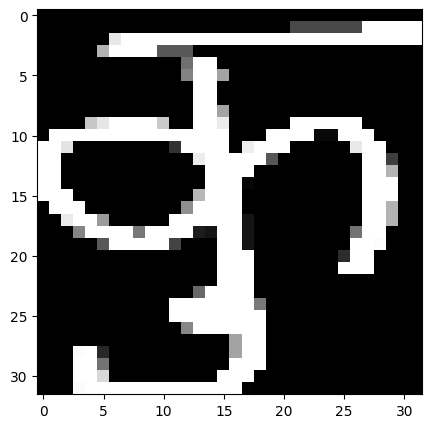
* Loading the Image: The system loads an input image containing Devanagari characters.
* Preprocessing: The input image undergoes preprocessing, including Gaussian blur and thresholding, to enhance the contrast and prepare it for segmentation.
* Segmentation: The preprocessed image is segmented into individual characters using horizontal and vertical segmentation techniques. This step divides the image into smaller segments, each containing a single character.
* Localization: The segmented characters are localized on the original image to highlight their positions. This step helps visualize the boundaries of each character within the image.
* Character Classification: Each localized character is classified using a trained Convolutional Neural Network (CNN) model. The model predicts the character represented by each segment based on its features.
* Output: The system outputs the classified characters along with their positions in the original image. It also provides the accuracy of the classification algorithm.
* Visualization: The system can display the segmented characters, their positions in the original image, and any other relevant visualizations to aid in understanding the recognition process.

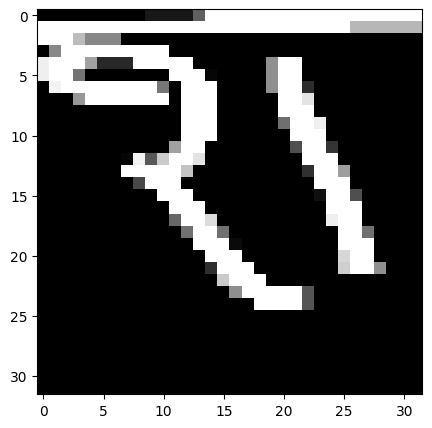
Overall, the OCR system provides an automated way to recognize and classify Devanagari characters in images, which can be useful for various applications requiring text extraction and analysis.

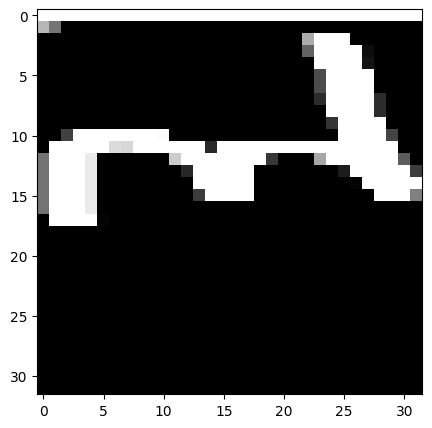
**RESULTS:-**

****

****

****

****

****

**('कु श ल', 99.3064820766449)[accuracy]**