Cam-scanner using OpenCV

Niladri Ghosh
ID-B2430100
Department of Computer Science

April 10, 2025

Abstract

This report presents a computer vision-based document scanner using OpenCV. The pipeline involves detecting the boundary of a document from an image, applying a perspective transform, and enhancing the resulting image to resemble a scanned copy. This project simulates the core functionality of mobile scanning apps using simple yet effective image processing techniques.

1 Introduction

Scanning documents with a smartphone has become common practice. This project implements a custom scanner using Python and OpenCV. The main steps include:

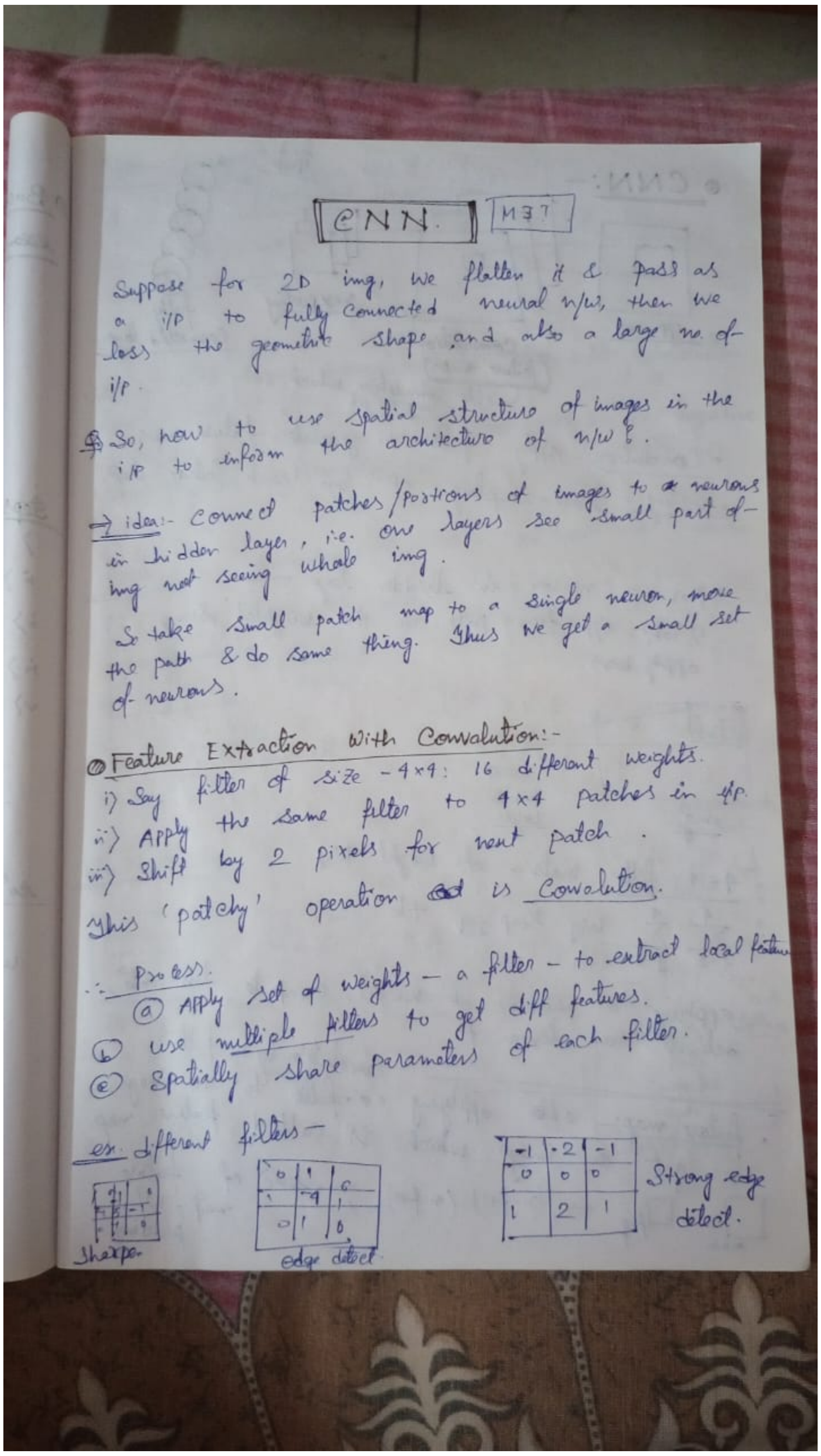
- Edge detection using Canny and contour approximation.
- Perspective transformation to ”flatten” the image.
- Image enhancement using adaptive thresholding.

2 Methodology

2.1 Step 1: Preprocessing

The input image is resized and converted to grayscale for noise reduction and processing efficiency.

```
image = cv2.imread('doc.jpg')
image = cv2.resize(image, (1300,800))
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
blurred = cv2.GaussianBlur(gray, (5, 5), 0)
```



CNN. M37

Suppose for 2D img, we flatten it & pass as a i/p to fully connected neural n/w, then we loss the geometric shape and also a large no. of i/p.

So, how to use spatial structure of images in the i/p to inform the architecture of n/w?

⇒ idea:- Connect patches/portions of images to neurons in hidden layer, i.e. our layers see small part of img not seeing whole img.

So take small patch map to a single neuron, move the path & do same thing. Thus we get a small set of neurons.

Feature Extraction with Convolution:-

i) Say filter of size - 4×4: 16 different weights.
ii) Apply the same filter to 4×4 patches in i/p.
iii) Shift by 2 pixels for next patch.

This 'patchy' operation ~~[illegible]~~ is Convolution.

∴ Process.
ⓐ Apply set of weights – a filter – to extract local features
ⓑ use multiple filters to get diff features.
ⓒ spatially share parameters of each filter.

ex. different filters –

[illegible]

Sharpen

0	1	0
1	-4	1
0	1	0

edge detect

-1	-2	-1
0	0	0
1	2	1

Strong edge detect.

Figure 1: Original Input Image

2.2 Step 2: Edge Detection and Contour Finding

Canny edge detection is applied, and the largest contour approximated to four points is assumed to be the document boundary.

```
edged = cv2.Canny(blurred, 30, 50)
contours, _ = cv2.findContours(edged, cv2.RETR_LIST, cv2.CHAIN_APPROX_SIMPLE)
```

```
contours = sorted(contours, key=cv2.contourArea, reverse=True)[:5]
```

2.3 Step 3: Perspective Transform

The detected contour is used to compute a perspective transform that warps the image, giving a top-down view.

```
warped = cv2.warpPerspective(image, matrix, (width, height))
```

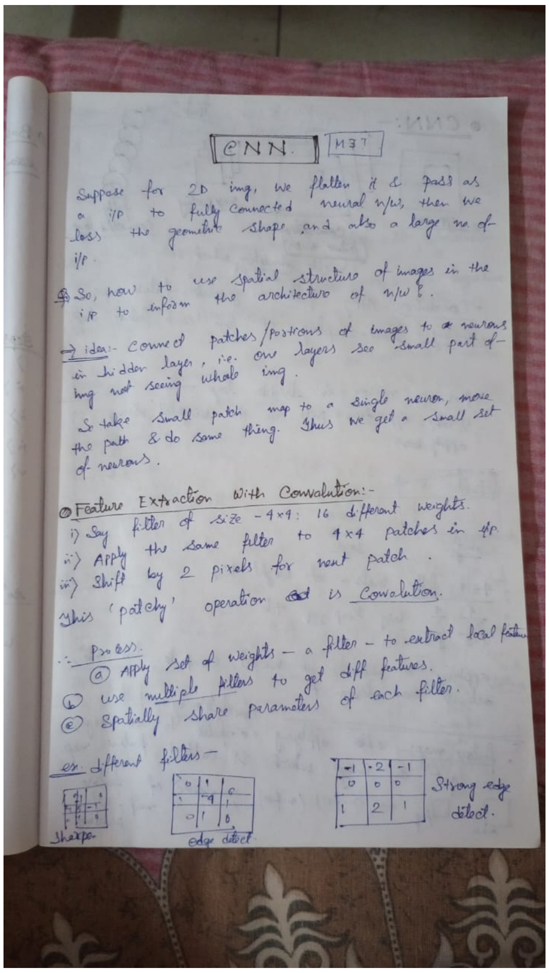
2.4 Step 4: Image Enhancement

To simulate the high-contrast appearance of scanned documents, adaptive thresholding is applied to the warped grayscale image.

```
final = cv2.adaptiveThreshold(
    gray_warped, 255, cv2.ADAPTIVE_THRESH_GAUSSIAN_C,
    cv2.THRESH_BINARY, 11, 2
)
```

3 Results

The implemented pipeline successfully detects and transforms a document image into a top-down, high-contrast scan-like output. The adaptive thresholding helps improve readability and mimics the appearance of a physical scanner output.



CNN. M37

Suppose for 2D img, we flatten it & pass as a i/p to fully connected neural n/w, then we loss the geometric shape and also a large no. of i/p.

So, how to use spatial structure of images in the i/p to inform the architecture of n/w?

→ idea:- connect patches/portions of images to a neurons in hidden layer, i.e. our layers see small part of img not seeing whole img.

So take small patch map to a single neuron, move the path & do same thing. Thus we get a small set of neurons.

Feature Extraction With Convalution:-

i) Say filter of size - 4×4: 16 different weights.

ii) Apply the same filter to 4×4 patches in i/p.

iii) Shift by 2 pixels for next patch.

This 'patchy' operation is Convolution.

∴ Process:- a filter - to extract local featu

(a) Apply set of weights - a filter - to extract local featu

(b) use multiple filters to get diff features.

(c) Spatially share parameters of each filter.

ex. different filters -

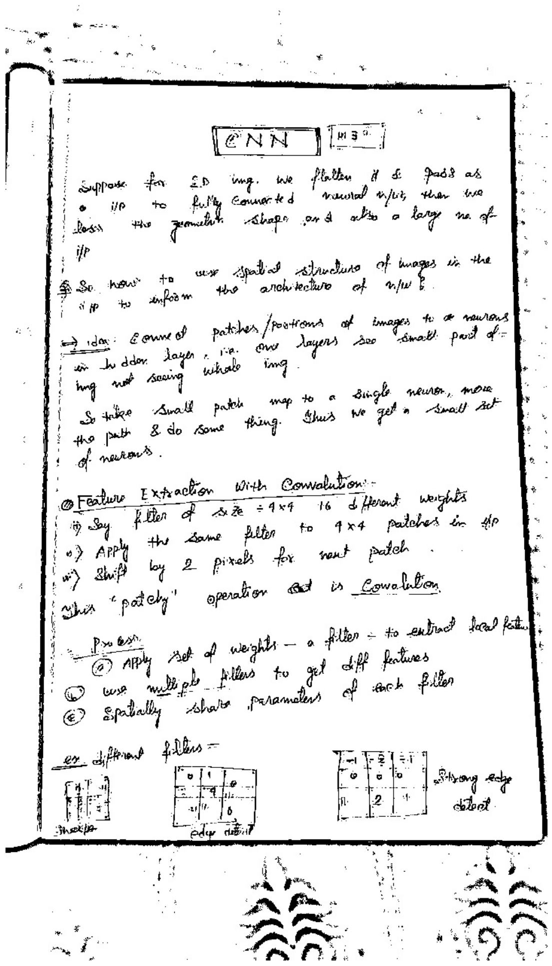
Sharpen

edge detect

-1	-2	-1
0	0	0
1	2	1

Strong edge detect.

(a) Left Image



CNN M37

Suppose for 2D img, we flatten it & pass as a i/p to fully connected neural n/w, then we loss the geometric shape and also a large no. of i/p.

So, how to use spatial structure of images in the i/p to inform the architecture of n/w?

→ idea:- connect patches/portions of images to a neurons in hidden layer, i.e. our layers see small part of img not seeing whole img.

So take small patch map to a single neuron, move the path & do same thing. Thus we get a small set of neurons.

Feature Extraction With Convalution:-

i) Say filter of size - 4×4: 16 different weights.

ii) Apply the same filter to 4×4 patches in i/p.

iii) Shift by 2 pixels for next patch.

This "patchy" operation is Convolution.

Process

(a) Apply set of weights - a filter - to extract local featu

(b) use multiple filters to get diff features

(c) Spatially share parameters of each filter

ex. different filters -

Sharpen

edge detect

Strong edge detect

(b) Right Image

Figure 2: Cam-scanner using OpenCV

4 Conclusion

This project demonstrates a simple but effective way to simulate a document scanner using OpenCV. The method is robust to perspective and lighting variations, making it suitable for practical use.

5 Future Enhancements

- Add support for automatic document detection in a video stream.
- Implement text extraction using OCR (e.g., Tesseract).
- Improve lighting normalization for better scan quality.