**DETECTS AND EXTRACTS TEXT FROM ANY IMAGE USING IMAGE TO TEXT API WITH IBM CLOUD**

**1.INTRODUCTION:-**

**1.1.OVERVIEW:-**

Text recognition in images is an active research area which attempts to develop a computer application with the ability to automatically read the text from images. Nowadays there is a huge demand of storing the information available on paper documents in to a computer readable form for later use. One simple way to store information from these paper documents in to computer system is to first scan the documents and then store them as images. However to reuse this information it is very difficult to read the individual contents and searching the contents form these documents line-by-line and word-by-word. The challenges involved are: font characteristics of the characters in paper documents and quality of the images. Due to these challenges, computer is unable to recognize the characters while reading them. Thus, there is a need of character recognition mechanisms to perform document image analysis which transforms documents in paper format to electronic format.

**1.2.PURPOSE:-**

This method is used to detect the text from Image . This might be computer written or hand written using image to text API with IBM Cloud.Extracted text means retrieving text from image in the file format or normal lines by using some software like flask.This is very useful to extract text from the image.

**2.LITERATURE SURVEY:-**

**2.1.Existing Problem:-**

In this digital era, it isn’t uncommon to face the need to extract text from an image to make it editable. This is especially true due to our dependence on paper documents that can only be made digitally editable with OCR software’s help. [Optical character recognition](https://en.wikipedia.org/wiki/Optical_character_recognition" \t "_blank) (OCR) is a pattern recognition AI-based technology to identify text inside an image and turn it into an editable digital document. many tools let you use OCR technology to extract text from images. Whether you want to convert the images to text on a PC, phone, or online, there is a tool for it.Use google docs to extract text from image . On the desktop version of Google Docs, you can upload an image and use built-in OCR technology to extract text from an image. On the desktop version of Google Docs, you can upload an image and use built-in OCR technology to extract text from an image.

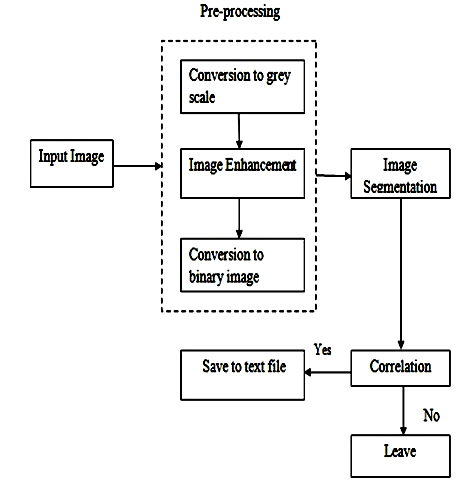
**2.2.Proposed System:-**

we are introducing our project which we can extract text easily. First the user should copy the url of that particular image and copy it and user should paste the url in our proposed project application. If they enter url of the image and click on submit.After submit in backend it uses the text to image rapid api to conversion of image to text.after that text in that image is displayed.if user not entered url properly its shows key error..

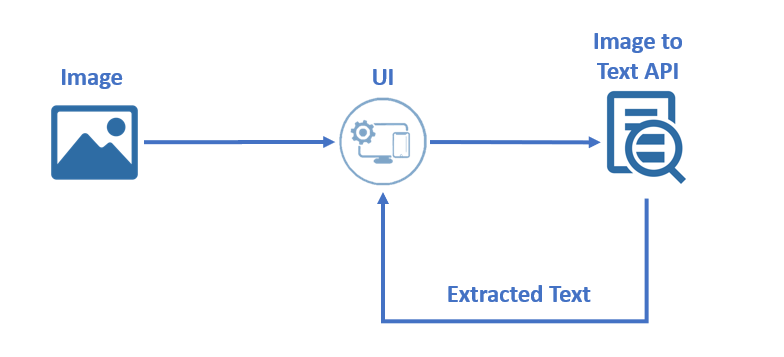
**3.THEORITICAL ANALYSIS:-**

**3.1.BLOCK DIAGRAM:-**

We are introducing a block diagram for better understanding purpose about our proposed project.By using the block digram we can understand it easily.The user enters the link in the html interface as a front end.In backend the html links with the python by using flask .If the user enters the correct input only the output will print correctly otherwise the output not prints correct way.

****

**3.2.TECHNICAL ARCHITECTURE:-**

****

**3.3.Hardware/Software Designing:-**

**Software Requirements:-**

* OS – Windows XP,7,8,10
* Jupyter Software
* User languages:HTML,Ajax
* Spyder Software
* Anaconda Command Prompt
* Flask Frame Work

**Hardware Components:**

* Processor – i3
* Hard Disk Storage – 10 GB
* RAM – 1GB

**4.EXPERIMENTAL INVESTIGATIONS:-**

In [1], images with colorful background is considered and a preprocessing method is described which improves upon the performance of the API and Flask. Here first text segmentation is done to separate the text from the colorful background by dividing the original image into k images. Then a classifier recognizes the image containing text. There was an improvement of about 20% compared to the API and flask performance by employing preprocessing.

Work by S. Akopyan, O.V. Belyaeva, T.P. Plechov and

D.Y. Turdakov [2] is based on a text extraction pipeline which is used to extract text from varied quality of images obtained from social media. Their work mainly focuses on dividing the input images into various classes and then preprocessing is done depending on the classes. This is followed by text recognition using Flask and API engine. The dataset collected from the social media is made use of in this work.

**5.RESULTS:-**

By using this application finally user can get the results of the text from the image.

**6.ADVANTAGES AND DISADVANTAGES:-**

**ADVANTAGES:-**

* Tilt text is detected
* High accuracy in natural scene
* Requires less text extraction database
* Most relevant and accurate is retrieved from the web

**DISADVANTAGES:-**

* Handwritten text cannot be accurately recognized

**7.APPLICATION:-**

* Analysis of documents can be easily done
* Industrial automation

**8.CONCLUSION:-**

By using this application finally user can get the results of the text from the image.In this model user can understand the code easily,and implementation easily

**9.FUTURE SCOPE:-**

In future we can include more features to our application so it will useful to our project.

**10.BIBILOGRAPHY:-**

1. Matteo Brisinello, Ratko Grbi, Dejan Stefanovi and Robert Pekai- Kova, Optical Character Recognition on images with colorful background, 2018, IEEE 8th International Conference on Consumer Electronics – Berlin (ICCE-Berlin).
2. M.S. Akopyan, O.V. Belyaeva, T.P. Plechov and D.Y. Turdakov, Text recognition on images from social media, 2019, Ivannikov Memorial Workshop (IVMEM).
3. Neha Agrawal, Arashdeep Kaur, An Algorithmic Approach for Text Recognition from Printed/Typed Text Images, 2018, 8th International Conference on Cloud Computing, Data Science & Engineering.
4. K. Karthick, K.B. Ravindrakumar, R. Francis, S. Ilankannan, Steps Involved in Text Recognition and Recent Research in OCR; A Study, International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277-3878, Volume-8, Issue-1, May 2019.
5. Anupriya Shrivastava, Amudha J., Deepa Gupta, Kshitij Sharma, Deep Learning Model for Text Recognition in Images, 10th ICCCNT 2019 July 6-8, 2019, IIT – Kanpur, Kanpur, India.
6. Pratik Madhukar Manwatkar, Dr. Kavita R. Singh, A Technical Review on Text Recognition from Images, IEEE Sponsored 9th International Conference on Intelligent Systems and Control (ISCO), 2015.

**SOURCE CODE:-**

# -\*- coding: utf-8 -\*-

"""

Created on Thu Nov 5 23:34:10 2020

@author: Madhav Rathi

"""

from flask import Flask, request, render\_template

import numpy as np

import re

import os

from gevent.pywsgi import WSGIServer

import requests

app = Flask(\_\_name\_\_)

def check(output):

url = "https://image-to-text2.p.rapidapi.com/cloudVision/imageToText"

querystring = {"source":output,"sourceType":"url"}

payload = '''{\r\n \"source\": "'''+output+'''" ,\r\n \"sourceType\": \"url\"\r\n}'''

headers = {

'content-type': "application/json",

'x-rapidapi-key': "fbd4199c1amsh2c93fd54d73d474p1d7c4cjsn786fe34bf93d",

'x-rapidapi-host': "image-to-text2.p.rapidapi.com"

}

response = requests.request("POST", url, data=payload, headers=headers, params=querystring)

print(response.text)

return (response.json()['text'])

#home page

@app.route('/')

def home():

return render\_template('base.html')

#Summarizer page

@app.route('/predict',methods=['POST'])

def predict():

output=request.form['output']

text=check(output)

return render\_template('base.html',output=text)

port = os.getenv('VCAP\_APP\_PORT','8080')

if \_\_name\_\_ == "\_\_main\_\_":

app.secret\_key = os.urandom(12)

app.run(debug=True,host='0.0.0.0',port=port)

**SOURCE CODE-HTML:-**

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Information Extractor</title>

<link href="https://cdn.bootcss.com/bootstrap/4.0.0/css/bootstrap.min.css" rel="stylesheet">

<link href='https://fonts.googleapis.com/css?family=Montserrat' rel='stylesheet'>

<link href='https://fonts.googleapis.com/css?family=Josefin Sans' rel='stylesheet'>

<script src="https://cdn.bootcss.com/popper.js/1.12.9/umd/popper.min.js"></script>

<script src="https://cdn.bootcss.com/jquery/3.3.1/jquery.min.js"></script>

<script src="https://cdn.bootcss.com/bootstrap/4.0.0/js/bootstrap.min.js"></script>

<style>

.bg-dark3 {

background-color:#323746;

}

#result {

color: #0a1c4ed1;

}

.img-preview {

width: 256px;

height: 256px;

position: relative;

border: 5px solid #F8F8F8;

box-shadow: 0px 2px 4px 0px rgba(0, 0, 0, 0.1);

margin-top: 1em;

margin-bottom: 1em;

}

.img-preview>div {

width: 100%;

height: 100%;

background-size: 256px 256px;

background-repeat: no-repeat;

background-position: center;

}

input[type="file"] {

display: none;

}

.upload-label{

display: inline-block;

padding: 12px 30px;

background: #323746;

color: #fff;

font-size: 1em;

transition: all .4s;

cursor: pointer;

font-weight:bold;

}

.upload-label:hover{

background: #5c5c8d;

color: #fff;

font-weight:bold;

}

.loader {

border: 8px solid #f3f3f3; /\* Light grey \*/

border-top: 8px solid #323746; /\* Blue \*/

border-radius: 50%;

width: 50px;

height: 50px;

animation: spin 1s linear infinite;

}

@keyframes spin {

0% { transform: rotate(0deg); }

100% { transform: rotate(360deg); }

}

</style>

</head>

<body style="font-family:Montserrat;">

<nav class="navbar navbar-dark bg-dark3">

<div class="container" style="font-size:30px;color:white;font-family: 'Josefin Sans';">

Information Extractor from Images

</div>

</nav>

<div class="container">

<div id="content" style="margin-top:2em;">

<div class="container">

<div class="row">

<div class="col-sm-6 bd" style="padding-right:6%;">

<h3><b>Information Extractor</b></h3>

<br>

<p style="text-align:justify;">Came accross a banner or any picture with text or a visting card? Would you like to store the information in that image as a text for future use? Seems difficult!! This application helps you do that. Browse the image, paste the URL of the image get the text extracted.</p>

<img src="https://todologix.com/wp-content/uploads/2019/04/illustration\_02-1.svg" style="height:450px;width:550px"class="img-rounded" alt="Gesture">

</div>

<div class="col-sm-6">

<div>

<h4>Drop in the image URL to extract the text!</h4><br>

<form action = "http://localhost:5000/predict" id="upload-file" method="post" enctype="multipart/form-data">

<input type="text" name="output" placeholder="Specify the image URL" style="width:100%;border: 1px solid black; border-radius:4px;height:6%;">

<br><br>

<input type="submit" style="background: #323746;background-border:#323746;color:white;" value="Submit">

<!--label for="imageUpload" class="upload-label">

Choose...

</label-->

<!--input type="file" name="image" id="imageUpload" accept=".png, .jpg, .jpeg"-->

</form>

<div class="loader" style="display:none;"></div>

<h3>

<span id="result" style="text-align:justify;"><center><b>The text extracted is:</b></center><br>{{output}} </span>

</h3>

</div>

</div>

</div>

</div>

</div>

</div>

</body>

<footer>

<!--script>

$(document).ready(function () {

// Init

$('.image-section').hide();

$('.loader').hide();

$('#result').hide();

// Upload Preview

function readURL(input) {

if (input.text && input.text[0]) {

var reader = new FileReader();

reader.onload = function (e) {

$('#imagePreview').attr('src', e.target.result + );

$('#imagePreview').hide();

$('#imagePreview').fadeIn(650);

}

reader.readAsDataURL(input.files[0]);

}

}

$("#imageUpload").change(function () {

$('.image-section').show();

$('#btn-predict').show();

$('#result').text('');

$('#result').hide();

readURL(this);

});

// Predict

$('#btn-predict').click(function () {

var form\_data = new FormData($('#upload-file')[0]);

// Show loading animation

$(this).hide();

$('.loader').show();

// Make prediction by calling api /predict

$.ajax({

type: 'POST',

url: '/predict',

data: form\_data,

contentType: false,

cache: false,

processData: false,

async: true,

success: function (data) {

// Get and display the result

$('.loader').hide();

$('#result').fadeIn(600);

$('#result').text('Prediction : '+data);

console.log('Success!');

},

});

});

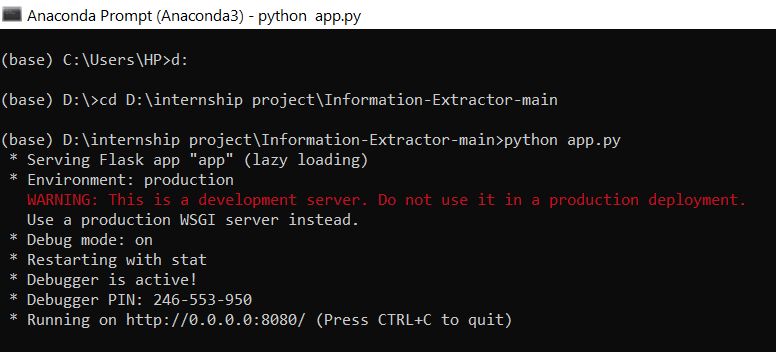
});

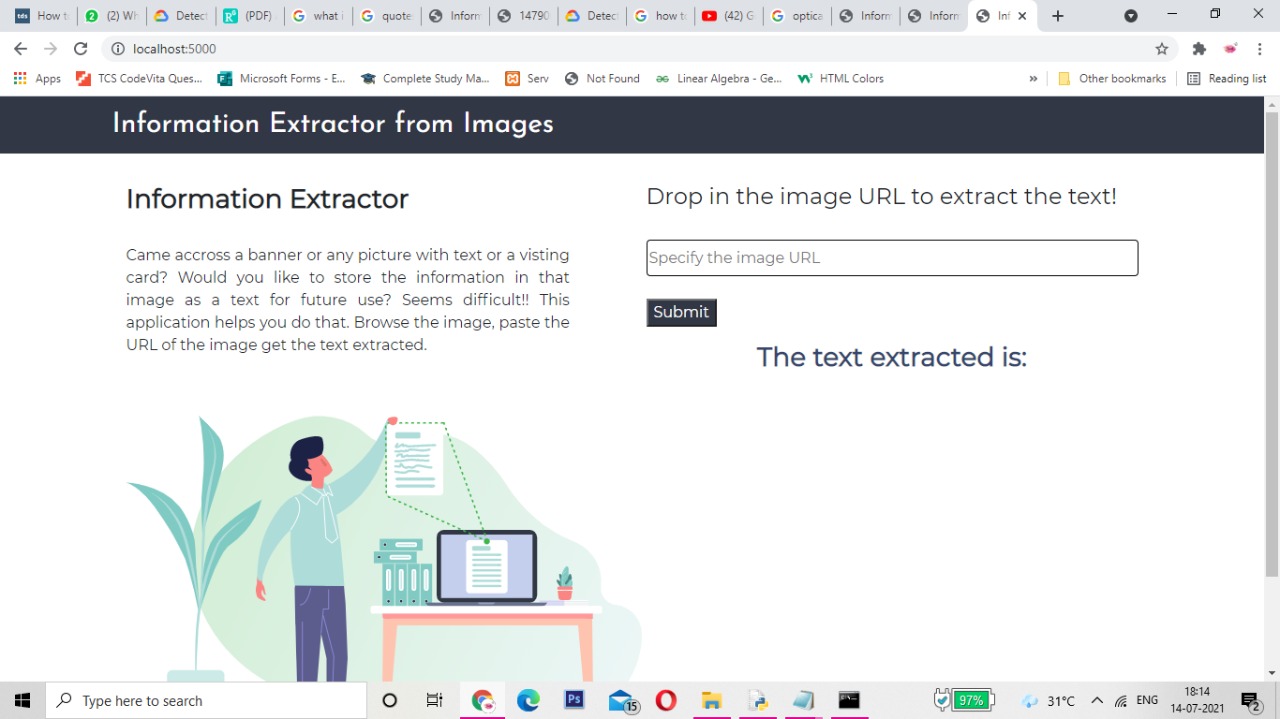
</script-->

</footer>

</html>

OUTPUT:-

****

****



