使用 Python 的图像隐写术

原创

小白学视觉 于 2021-09-02 10:05:00 发布 878 收藏 3
文章标签： python 人工智能 java 编程语言 机器学习

本文链接：https://blog.csdn.net/qq_42722197/article/details/120073039

重磅干货，第一时间送达

今天，世界正在见证前所未有的数据爆炸，我们每天产生的数据量确实令人难以置信。福布斯的文章“我们每天创造多少数据？”指出，以我们目前的速度每天创建大约2.5 千亿字节的数据，但随着物联网 (IoT) 的发展，这一速度只会加快。仅在过去两年中，世界上 90% 的数据都是生成的，这篇值得重读！

现代计算世界围绕着数据这个词，但到底是什么让人如此感兴趣呢？在当今世界，企业已经开始意识到数据就是力量，因为它可以预测客户趋势、增加销售额，并将公司推向新的高度。随着技术的快速进步和数据不断创新，数据安全已成为我们的首要任务。随着每天数以千计的消息和数据在互联网上从一个地方传输到另一个地方，数据共享正在增加，数据的保护是发送者的主要关注点，我们必须以只有接收者才能理解的秘密方式对消息进行加密，这一点非常重要。

在本文中，我们将了解什么是最低有效位隐写术，以及我们如何使用 python 实现它。

什么是隐写术？

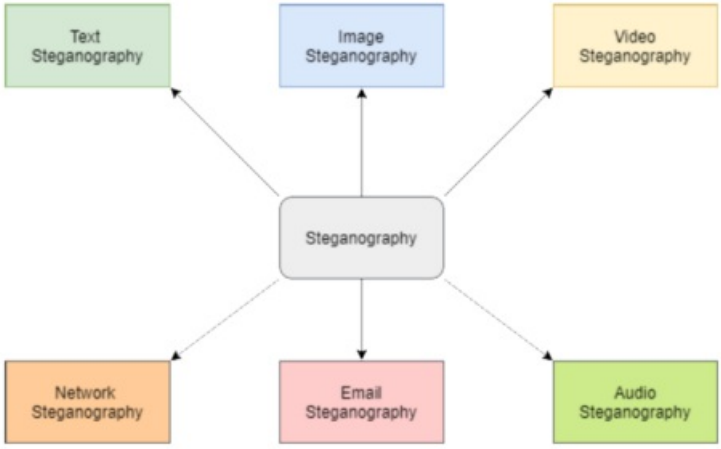
隐写术是将秘密信息隐藏在更大的信息中的过程，使人无法知道隐藏信息的存在或内容。隐写术的目的是保持双方之间的秘密通信，与隐藏秘密消息内容的密码学不同，隐写术隐藏了消息传递这一事实。虽然隐写术不同于密码术，但两者之间有很多类比，一些作者将隐写术归类为密码术的一种形式，因为隐藏通信是一种秘密通信。

使用隐写术优于加密术？

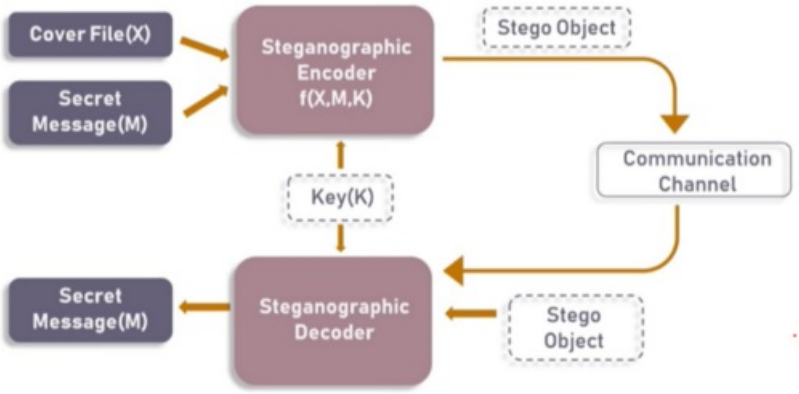
到目前为止，密码学一直在保护发送者和预期接收者之间的机密性方面发挥着最终作用。然而，如今除了密码学之外，隐写术被越来越多地用于为隐藏数据添加更多保护层。与单独使用密码学相比，使用隐写术的优势在于，预期的秘密消息本身不会引起人们的注意，不会成为审查的对象。显而易见的加密消息，无论它们多么牢不可破，都会引起人们的兴趣，并且在加密是非法的国家，这些信息本身可能会被指控有罪。[1]

隐写术的类型

隐写术工作已在不同的传输媒体上进行，如图像、视频、文本或音频。



基本隐写模型



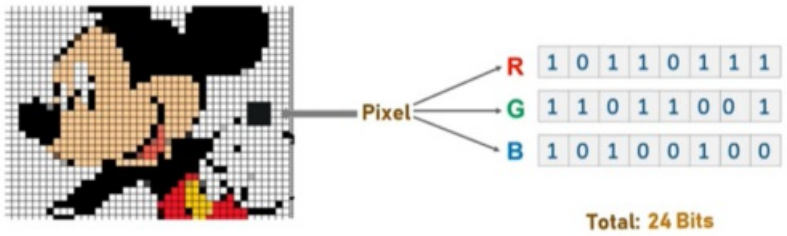
如上图所示，需要隐藏的原始图像文件 (X) 和秘密消息 (M) 都作为输入输入到隐写编码器中，隐写编码器函数 f(X,M,K) 通过使用最低有效位编码等技术将秘密消息嵌入到封面图像文件中，生成的隐藏图像看起来与封面图像文件非常相似，没有明显的变化，这样就完成了编码。为了检索秘密消息，隐写对象被输入到隐写解码器中。[3]

本文将帮助我们使用 Python 实现图像隐写术，它将帮助我们编写 Python 代码，使用一种称为最低有效位的技术隐藏文本消息。

最低有效位隐写术

我们可以将数字图像描述为一组有限的数字值，称为像素。像素是图像中最小的单个元素，其值表示给定颜色在任何特定点的亮度。因此，我们可以将图像视为包含固定行数和列数的像素矩阵（或二维数组）。

最低有效位 (LSB) 是一种技术，其中每个像素的最后一位被修改并替换为秘密消息的数据位。



从上图可以清楚地看出，如果我们改变 MSB 会对最终值产生更大的影响，但如果我们改变 LSB 对最终值的影响很小，因此我们使用最低有效位隐写术。

LSB 技术是如何工作的？

每个像素包含红、绿、蓝三个值，这些值的范围是 0 到 255，换句话说，它们是 8 位值。[4]

让我们举一个例子来说明这项技术的工作原理，假设我们想将消息“ hi ”隐藏到具有以下像素值的4x4图像中：

[(225, 12, 99), (155, 2, 50), (99, 51, 15), (15, 55, 22),(155, 61, 87), (63, 30, 17), (1, 55, 19), (99, 81, 66),(219, 77, 91), (69, 39, 50), (18, 200, 33), (25, 54, 190)]

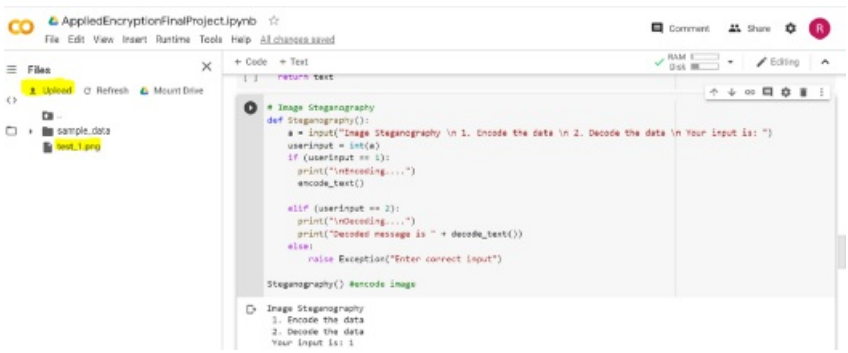
使用ASCII 表，我们可以将秘密消息转换为十进制值，然后转换为二进制值：0110100 0110101。现在，我们逐个迭代像素值，在将其转换为二进制后，我们将每个最低有效位依次替换为该消息位（例如 225 是 11100001，我们用第一个数据位（0）等替换最后一位，右边的位（1），依此类推）。这只会将像素值修改为 +1 或 -1，这根本不明显。执行LSBS后得到的像素值如下图：

[(224, 13, 99),(154, 3, 50),(98, 50, 15),(15, 54, 23),(154, 61, 87),(63, 30, 17),(1, 55, 19),(99, 81, 66),(219, 77, 91),(69, 39, 50),(18, 200, 33),(25, 54, 190)]

使用 Python 在图像中隐藏文本

在本节中，我们可以找到使用 Python 代码一步一步地隐藏和显示过程，打开一个google collab笔记本并按照以下步骤操作：

在开始编写代码之前，我们可以使用左侧菜单栏中显示的上传选项上传我们想用于隐写术的图像（png）。



第 1 步：导入所有必需的 Python 库

```
#import all the required libraries

import cv2
import numpy as np
import types
from google.colab.patches import cv2_imshow #Google colab crashes if you try to display
#image using cv2.imshow() thus use this import
```

第 2 步：定义一个将任意类型数据转换为二进制的函数，我们将在编码和解码阶段使用该函数来将秘密数据和像素值转换为二进制。

```
def messageToBinary(message):
  if type(message) == str:
    return ''.join([ format(ord(i), "08b") for i in message ])
  elif type(message) == bytes or type(message) == np.ndarray:
    return [ format(i, "08b") for i in message ]
  elif type(message) == int or type(message) == np.uint8:
    return format(message, "08b")
  else:
    raise TypeError("Input type not supported")
```

第 3 步：编写一个函数，通过改变 LSB 将秘密信息隐藏到图像中

```
# Function to hide the secret message into the image

def hideData(image, secret_message):

  # calculate the maximum bytes to encode
  n_bytes = image.shape[0] * image.shape[1] * 3 // 8
  print("Maximum bytes to encode:", n_bytes)

  #Check if the number of bytes to encode is less than the maximum bytes in the image
  if len(secret_message) > n_bytes:
      raise ValueError("Error encountered insufficient bytes, need bigger image or less data !!")

  secret_message += "#####" # you can use any string as the delimeter

  data_index = 0
  # convert input data to binary format using messageToBinary() fucntion
  binary_secret_msg = messageToBinary(secret_message)

  data_len = len(binary_secret_msg) #Find the length of data that needs to be hidden
  for values in image:
```

```
for values in image:
    for pixel in values:
        # convert RGB values to binary format
        r, g, b = messageToBinary(pixel)
        # modify the least significant bit only if there is still data to store
        if data_index < data_len:
            # hide the data into least significant bit of red pixel
            pixel[0] = int(r[:-1] + binary_secret_msg[data_index], 2)
            data_index += 1
        if data_index < data_len:
            # hide the data into least significant bit of green pixel
            pixel[1] = int(g[:-1] + binary_secret_msg[data_index], 2)
            data_index += 1
        if data_index < data_len:
            # hide the data into least significant bit of  blue pixel
            pixel[2] = int(b[:-1] + binary_secret_msg[data_index], 2)
            data_index += 1
        # if data is encoded, just break out of the loop
        if data_index >= data_len:
            break

return image
```

第 4 步：定义一个函数，用于解码隐藏图像中的隐藏消息

```
[ ]  def showData(image):

       binary_data = ""
       for values in image:
           for pixel in values:
               r, g, b = messageToBinary(pixel) #convert the red,green and blue values into binary format
               binary_data += r[-1] #extracting data from the least significant bit of red pixel
               binary_data += g[-1] #extracting data from the least significant bit of red pixel
               binary_data += b[-1] #extracting data from the least significant bit of red pixel
       # split by 8-bits
       all_bytes = [ binary_data[i: i+8] for i in range(0, len(binary_data), 8) ]
       # convert from bits to characters
       decoded_data = ""
       for byte in all_bytes:
           decoded_data += chr(int(byte, 2))
           if decoded_data[-5:] == "#####": #check if we have reached the delimeter which is "#####"
               break
       #print(decoded_data)
       return decoded_data[:-5] #remove the delimeter to show the original hidden message
```

第 5 步：该函数将输入的图像名称和秘密消息作为用户的输入，并调用 hideData() 对消息进行编码

```
[ ]  # Encode data into image
     def encode_text():
       image_name = input("Enter image name(with extension): ")
       image = cv2.imread(image_name) # Read the input image using OpenCV-Python.
       #It is a library of Python bindings designed to solve computer vision problems.

       #details of the image
       print("The shape of the image is: ",image.shape) #check the shape of image to calculate the number of bytes in it
       print("The original image is as shown below: ")
       resized_image = cv2.resize(image, (500, 500)) #resize the image as per your requirement
       cv2_imshow(resized_image) #display the image


       data = input("Enter data to be encoded : ")
       if (len(data) == 0):
         raise ValueError('Data is empty')

       filename = input("Enter the name of new encoded image(with extension): ")
       encoded_image = hideData(image, data) # call the hideData function to hide the secret message into the selected image
       cv2.imwrite(filename, encoded_image)
```

第6步：创建一个函数，让用户输入需要解码的图片名称，调用showData()函数返回解码后的信息

```
# Decode the data in the image
def decode_text():
  # read the image that contains the hidden image
  image_name = input("Enter the name of the steganographed image that you want to decode (with extension) :")
  image = cv2.imread(image_name) #read the image using cv2.imread()

  print("The Steganographed image is as shown below: ")
  resized_image = cv2.resize(image, (500, 500))  #resize the original image as per your requirement
  cv2_imshow(resized_image) #display the Steganographed image

  text = showData(image)
  return text
```

第 7 步：主函数（）

```
# Image Steganography
def Steganography():
    a = input("Image Steganography \n 1. Encode the data \n 2. Decode the data \n Your input is: ")
    userinput = int(a)
    if (userinput == 1):
      print("\nEncoding....")
      encode_text()

    elif (userinput == 2):
      print("\nDecoding....")
      print("Decoded message is " + decode_text())
    else:
        raise Exception("Enter correct input")

Steganography() #encode image
```

结果：

对消息进行编码：

```
Image Steganography
 1. Encode the data
 2. Decode the data
 Your input is: 1

Encoding....
Enter image name(with extension): test_1.png
The shape of the image is:  (1254, 1254, 3)
The original image is as shown below:
```

```
Enter data to be encoded : hakunamatata
Enter the name of new encoded image(with extension): test_1_encoded.png
Maximum bytes to encode: 589693
```

解码消息：

```
Image Steganography
  1. Encode the data
  2. Decode the data
  Your input is: 2

Decoding....
Enter the name of the steganographed image that you want to decode (with extension) :test_1_encoded.png
The Steganographed image is as shown below:
```

```
Decoded message is hakunamatata
```

如果小伙伴们对代码感兴趣，可以在Github上找到我的笔记本。

参考：

[1].https://towardsdatascience.com/steganography-hiding-an-image-inside-another-77ca66b2acb1

[2].https://www.edureka.co/blog/steganography-tutorial

[3].https://www.forbes.com/sites/bernardmarr/2018/05/21/how-much-data-do-we-create-every-day-the-mind-blowing-stats-everyone-should-read/#191d0b0160ba

[4].https://www.ukessays.com/essays/computer-science/steganography-uses-methods-tools-3250.php

[5].https://www.thepythoncode.com/article/hide-secret-data-in-images-using-steganography-python

[6].https://www.youtube.com/watch?v=xepNoHgNj0w&t=1922s

Github代码连接：

https://github.com/rroy1212/Image_Steganography/blob/master/ImageSteganography.ipynb

下载1：OpenCV-Contrib扩展模块中文版教程

在「**小白学视觉**」公众号后台回复：**扩展模块中文教程**，即可下载全网第一份OpenCV扩展模块教程中文版，涵盖**扩展模块安装、SFM算法、立体视觉、目标跟踪、生物视觉、超分辨率处理**等二十多章内容。

下载2：Python视觉实战项目52讲

在「**小白学视觉**」公众号后台回复：**Python视觉实战项目**，即可下载包括**图像分割、口罩检测、车道线检测、车辆计数、添加眼线、车牌识别、字符识别、情绪检测、文本内容提取、面部识别**等31个视觉实战项目，助力快速学校计算机视觉。

下载3：OpenCV实战项目20讲

在「**小白学视觉**」公众号后台回复：**OpenCV实战项目20讲**，即可下载含有**20**个基于**OpenCV**实现20个**实战项目**，实现OpenCV学习进阶。

交流群

欢迎加入公众号读者群一起和同行交流，目前有SLAM、三维视觉、传感器、自动驾驶、计算摄影、检测、分割、识别、医学影像、GAN、算法竞赛等微信群（以后会逐渐细分），请扫描下面微信号加群，备注：”昵称+学校/公司+研究方向“，例如：”张三 + 上海交大 + 视觉SLAM“。请按照格式备注，否则不予通过。添加成功后会根据研究方向邀请进入相关微信群。**请勿**在群内发送广告，否则会请出群，谢谢理解~