在OpenCV中基於深度學習的邊緣檢測

小白 小白學視覺 2022-03-21 10:05

本文轉自：AI算法與圖像處理

導讀

分析了Canny的優劣，並給出了OpenCV使用深度學習做邊緣檢測的流程，文末有代碼鏈接。

在這篇文章中，我們將學習如何在OpenCV中使用基於深度學習的邊緣檢測，它比目前流行的canny邊緣檢測器更精確。邊緣檢測在許多用例中是有用的，如視覺顯著性檢測，目標檢測，跟踪和運動分析，結構從運動，3D重建，自動駕駛，圖像到文本分析等等。

什麼是邊緣檢測?

邊緣檢測是計算機視覺中一個非常古老的問題，它涉及到檢測圖像中的邊緣來確定目標的邊界，從而分離感興趣的目標。最流行的邊緣檢測技術之一是Canny邊緣檢測，它已經成為大多數計算機視覺研究人員和實踐者的首選方法。讓我們快速看一下Canny邊緣檢測。

Canny邊緣檢測算法

1983年，John Canny在麻省理工學院發明了Canny邊緣檢測。它將邊緣檢測視為一個信號處理問題。其核心思想是，如果你觀察圖像中每個像素的強度變化，它在邊緣的時候非常高。

在下面這張簡單的圖片中，強度變化只發生在邊界上。所以，你可以很容易地通過觀察像素強度的變化來識別邊緣。

現在，看下這張圖片。強度不是恆定的，但強度的變化率在邊緣處最高。(微積分複習：變化率可以用一階導數(梯度)來計算。)

Canny邊緣檢測器通過4步來識別邊緣：

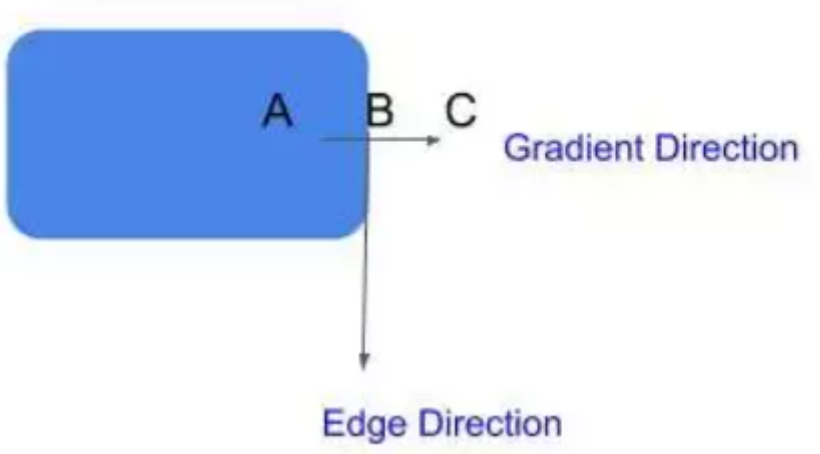
1. **去噪**：因為這種方法依賴於強度的突然變化，如果圖像有很多隨機噪聲，那麼會將噪聲作為邊緣。所以，使用5×5的高斯濾波器平滑你的圖像是一個非常好的主意。
2. **梯度計算**：下一步，我們計算圖像中每個像素的強度的梯度(強度變化率)。我們也計算梯度的方向。

$$Edge\_Gradient\ (G) = \sqrt{G_x^2 + G_y^2}$$

$$Angle\ (\theta) = \tan^{-1}\left(\frac{G_y}{G_x}\right)$$

梯度方向垂直於邊緣，它被映射到四個方向中的一個(水平、垂直和兩個對角線方向)。

3. **非極大值抑制**：現在，我們想刪除不是邊緣的像素(設置它們的值為0)。你可能會說，我們可以簡單地選取梯度值最高的像素，這些就是我們的邊。然而，在真實的圖像中，梯度不是簡單地在只一個像素處達到峰值，而是在臨近邊緣的像素處都非常高。因此我們在梯度方向上取3×3附近的局部最大值。

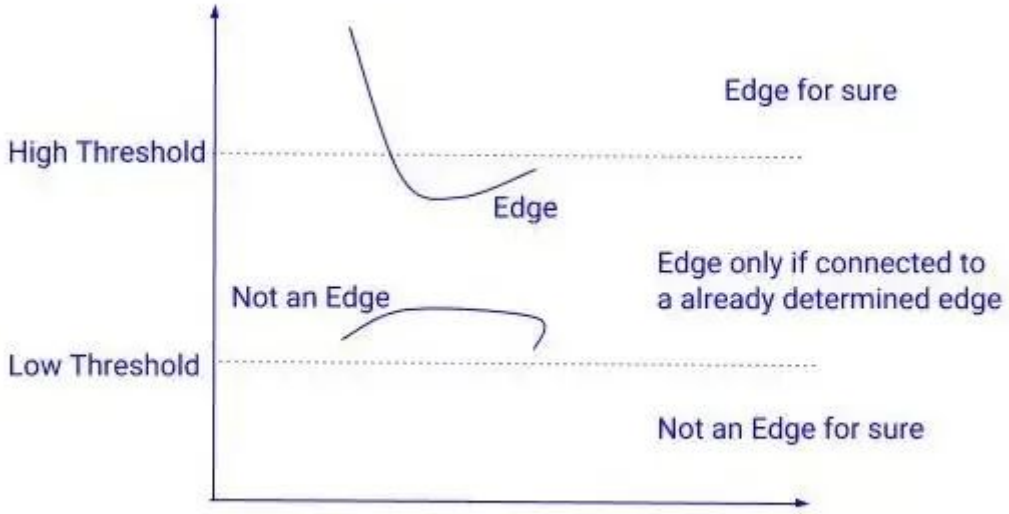


4. **遲滯閾值化**：在下一步中，我們需要決定一個梯度的閾值，低於這個閾值所有的像素都將被抑制(設置為0)。而Canny邊緣檢測器則採用遲滯閾值法。遲滯閾值法是一種非常簡單而有效的方法。我們使用兩個閾值來代替只用一個閾值：

高閾值= 選擇一個非常高的值，這樣任何梯度值高於這個值的像素都肯定是一個邊緣。

低閾值= 選擇一個非常低的值，任何梯度值低於該值的像素絕對不是邊緣。

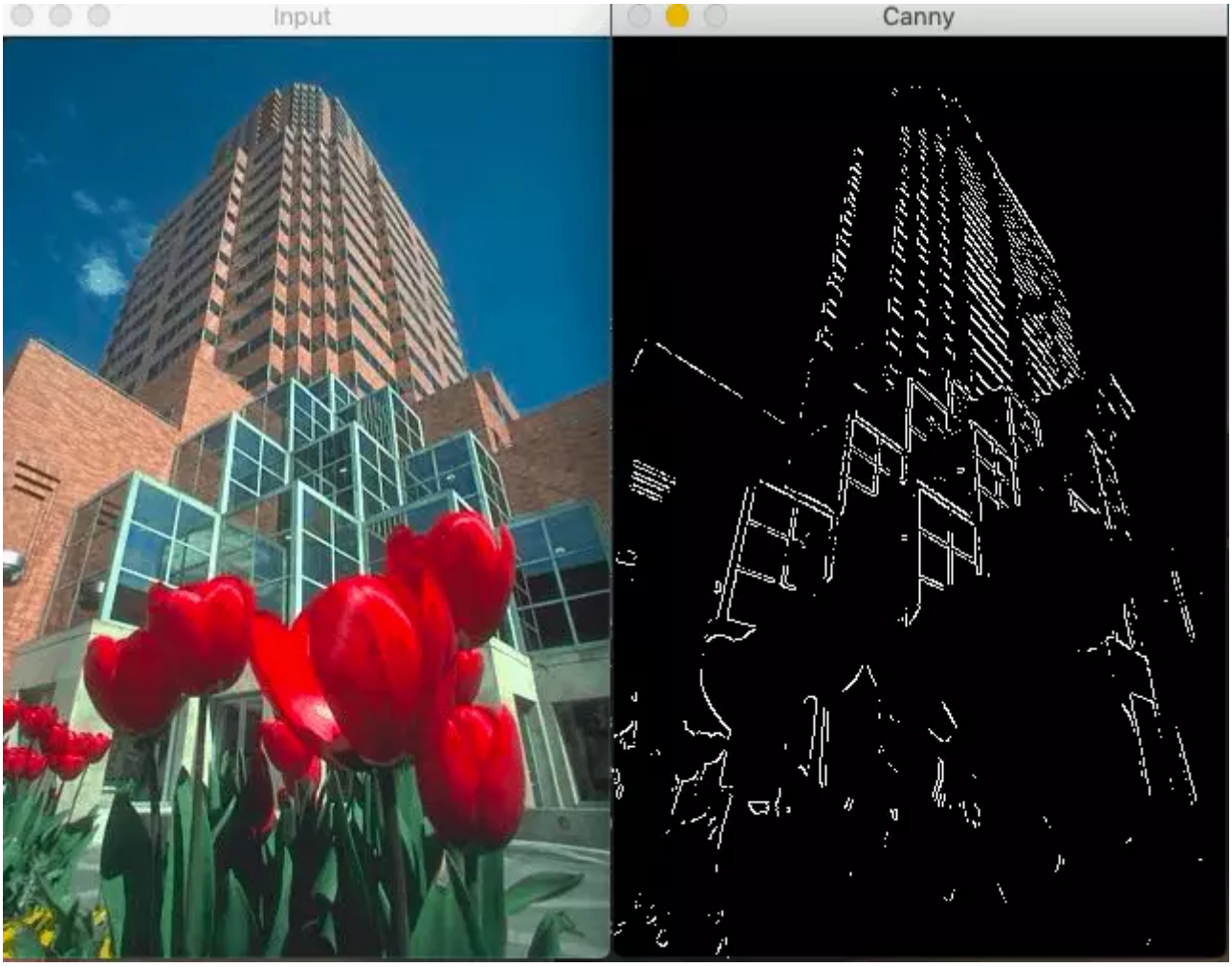
在這兩個閾值之間有梯度的像素會被檢查，如果它們和邊緣相連，就會留下，否則就會去掉。



遲滯閾值化

Canny 邊緣檢測的問題：

由於Canny邊緣檢測器只關注局部變化，沒有語義(理解圖像的內容)理解，精度有限(很多時候是這樣)。



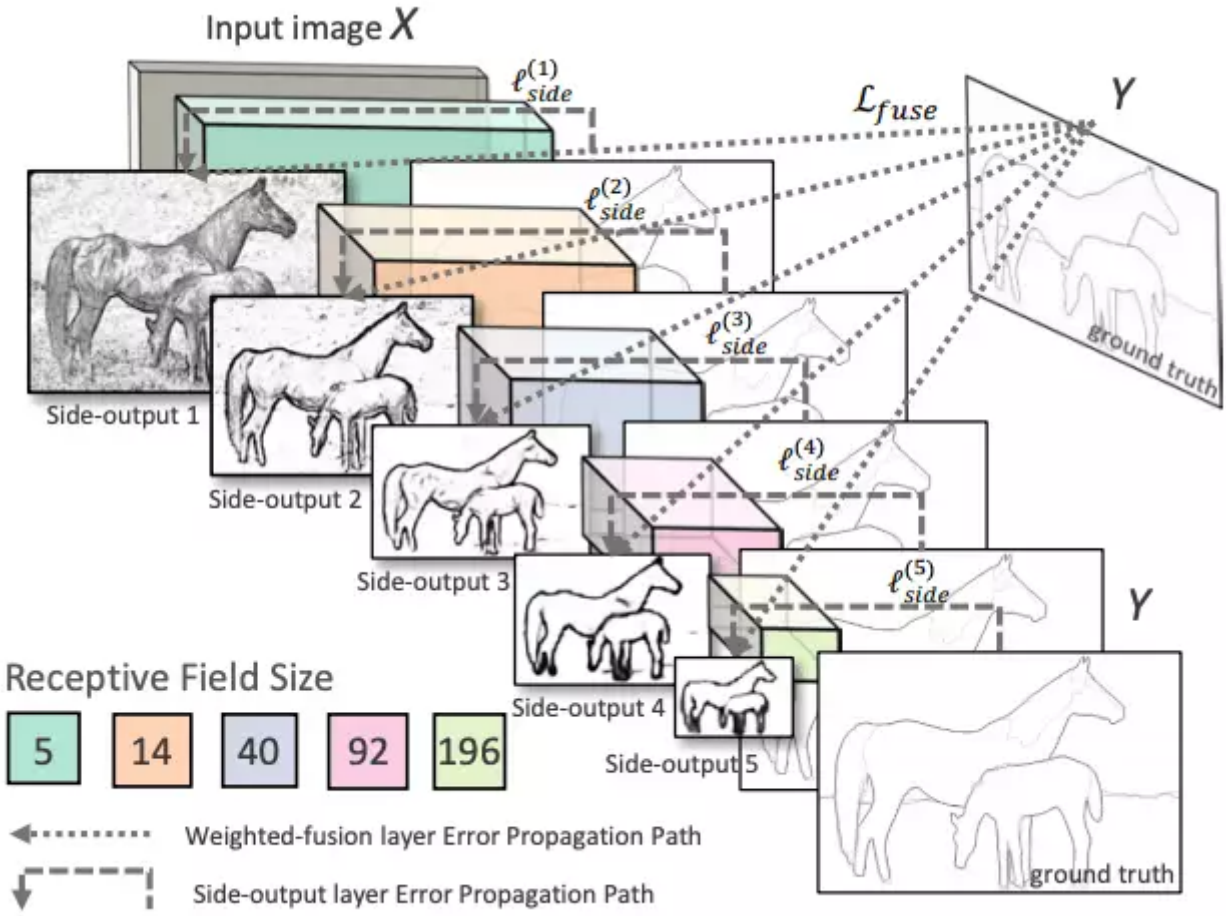
Canny邊緣檢測器在這種情況下會失敗，因為沒有理解圖像的上下文

語義理解對於邊緣檢測是至關重要的，這就是為什麼使用機器學習或深度學習的基於學習的檢測器比canny邊緣檢測器產生更好的結果。

OpenCV中基於深度學習的邊緣檢測

OpenCV在其全新的DNN模塊中集成了基於深度學習的邊緣檢測技術。你需要OpenCV 3.4.3或更高版本。這種技術被稱為整體嵌套邊緣檢測或HED，是一種基於學習的端到端邊緣檢測系統，使用修剪過的類似vgg的捲積神經網絡進行圖像到圖像的預測任務。

HED利用了中間層的輸出。之前的層的輸出稱為side output，將所有5個卷積層的輸出進行融合，生成最終的預測。由於在每一層生成的特徵圖大小不同，它可以有效地以不同的尺度查看圖像。



網絡結構：整體嵌套邊緣檢測

HED方法不僅比其他基於深度學習的方法更準確，而且速度也比其他方法快得多。這就是為什麼OpenCV決定將其集成到新的DNN模塊中。以下是這篇論文的結果：

Table 4. Results on BSDS500. ∗BSDS300 results,†GPU time

	ODS	OIS	AP	FPS
Human	.80	.80	-	-
Canny	.600	.640	.580	15
Felz-Hutt [9]	.610	.640	.560	10
BEL [5]	.660∗	-	-	1/10
gPb-owt-ucm [1]	.726	.757	.696	1/240
Sketch Tokens [24]	.727	.746	.780	1
SCG [31]	.739	.758	.773	1/280
SE-Var [6]	.746	.767	.803	2.5
OEF [13]	.749	.772	.817	-
DeepNets [21]	.738	.759	.758	1/5†
N4-Fields [10]	.753	.769	.784	1/6†
DeepEdge [2]	.753	.772	.807	$1/10^3$†
CSCNN [19]	.756	.775	.798	-
DeepContour [34]	.756	.773	.797	1/30†
HED (ours)	**.782**	**.804**	**.833**	2.5†, 1/12

在OpenCV中訓練深度學習邊緣檢測的代碼

OpenCV使用的預訓練模型已經在Caffe框架中訓練過了，可以這樣加載：

```
sh download_pretrained.sh
```

網絡中有一個crop層，默認是沒有實現的，所以我們需要自己實現一下。

```
class CropLayer(object):
    def __init__(self, params, blobs):
        self.xstart = 0
        self.xend = 0
        self.ystart = 0
        self.yend = 0

    # Our layer receives two inputs. We need to crop the first input blob
    # to match a shape of the second one (keeping batch size and number of c
    def getMemoryShapes(self, inputs):
        inputShape, targetShape = inputs[0], inputs[1]
        batchSize, numChannels = inputShape[0], inputShape[1]
        height, width = targetShape[2], targetShape[3]

        self.ystart = (inputShape[2] - targetShape[2]) // 2
        self.xstart = (inputShape[3] - targetShape[3]) // 2
        self.yend = self.ystart + height
        self.xend = self.xstart + width
```

```
        return [[batchSize, numChannels, height, width]]

    def forward(self, inputs):
        return [inputs[0][:,:,self.ystart:self.yend,self.xstart:self.xend]]
```

現在，我們可以重載這個類，只需用一行代碼註冊該層。

```
cv.dnn_registerLayer('Crop', CropLayer)
```

現在，我們準備構建網絡圖並加載權重，這可以通過OpenCV的dnn.readNe函數。

```
net = cv.dnn.readNet(args.prototxt, args.caffemodel)
```

現在，下一步是批量加載圖像，並通過網絡運行它們。為此，我們使用cv2.dnn.blobFromImage方法。該方法從輸入圖像中創建四維blob。

```
blob = cv.dnn.blobFromImage(image, scalefactor, size, mean, swapRB, crop)
```

其中：

image：是我們想要發送給神經網絡進行推理的輸入圖像。

scalefactor：圖像縮放常數，很多時候我們需要把uint8的圖像除以255，這樣所有的像素都在0到1之間。默認值是1.0，不縮放。

size：輸出圖像的空間大小。它將等於後續神經網絡作為blobFromImage輸出所需的輸入大小。

swapRB：布爾值，表示我們是否想在3通道圖像中交換第一個和最後一個通道。OpenCV默認圖像為BGR格式，但如果我們想將此順序轉換為RGB，我們可以將此標誌設置為True，這也是默認值。

mean：為了進行歸一化，有時我們計算訓練數據集上的平均像素值，並在訓練過程中從每幅圖像中減去它。如果我們在訓練中做均值減法，那麼我們必須在推理中應用它。這個平均值是一個對應於R, G, B通道的元組。例如Imagenet數據集的均值是R=103.93, G=116.77, B=123.68。如果我們使用swapRB=False，那麼這個順序將是(B, G, R)。

crop：布爾標誌，表示我們是否想居中裁剪圖像。如果設置為True，則從中心裁剪輸入圖像時，較小的尺寸等於相應的尺寸，而其他尺寸等於或大於該尺寸。然而，如果我們將其設置為False，它將保留長寬比，只是將其調整為固定尺寸大小。

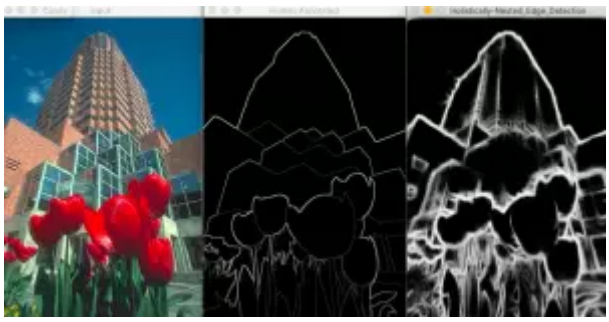
在我們這個場景下：

```
inp = cv.dnn.blobFromImage(frame, scalefactor=1.0, size=(args.width, args.he
                           mean=(104.00698793, 116.66876762, 122.67891434),
                           crop=False)
```

現在，我們只需要調用一下前向方法。

```
net.setInput(inp)
out = net.forward()
out = out[0, 0]
out = cv.resize(out, (frame.shape[1], frame.shape[0]))
out = 255 * out
out = out.astype(np.uint8)
out=cv.cvtColor(out,cv.COLOR_GRAY2BGR)
con=np.concatenate((frame,out),axis=1)
cv.imshow(kWinName,con)
```

結果：



中間的圖像是人工標註的圖像，右邊是HED的結果

中間的圖像是人工標註的圖像，右邊是HED的結果

文中的代碼：https://github.com/sankit1/cv-tricks.com/tree/master/OpenCV/Edge_detection

— END —

英文原文：https://cv-tricks.com/opencv-dnn/edge-detection-hed/

下載1：OpenCV-Contrib擴展模塊中文版教程

在「**小白學視覺**」公眾號後台回復：**擴展模塊中文教程，**即可下載全網第一份OpenCV擴展模塊教程中文版，涵蓋**擴展模塊安裝、SFM算法、立體視覺、目標跟踪、生物視覺、超分辨率處理**等二十多章內容。

下載2：Python視覺實戰項目52講

在「**小白學視覺**」公眾號後台回復：**Python視覺實戰項目，**即可下載包括**圖像分割、口罩檢測、車道線檢測、車輛計數、添加眼線、車牌識別、字符識別、情緒檢測、文本內容提取、面部識別**等31個視覺實戰項目，助力快速學校計算機視覺。

下載3：OpenCV實戰項目20講

在「**小白學視覺**」公眾號後台回復：**OpenCV實戰項目20講，**即可下載含有**20**個基於**OpenCV**實現20個**實戰項目**，實現OpenCV學習進階。

交流群

歡迎加入公眾號讀者群一起和同行交流，目前有**SLAM 、三維視覺、傳感器、自動駕駛、計算攝影**、檢測、分割、識別、**醫學影像、GAN 、算法競賽**等微信群（以後會逐漸細分），請掃描下面微信號加群，備註：”暱稱+學校/公司+研究方向“，例如：”張三 + 上海交大 + 視覺SLAM“。**請按照格式備註，否則不予通過。**添加成功後會根據研究方向邀請進入相關微信群。**請勿**在群內發送**廣告**，否則會請出群，謝謝理解~

喜歡此內容的人還喜歡

計算機視覺專家：如何從C++轉Python

小白學視覺