International Journal of Psychosocial Rehabilitation, Vol. 24, Issue 08, 2020
ISSN: 1475-7192

Image Colourization and Object Detection Using Convolutional Neural Networks

[1]K. Sree Kumar, [2]Sourya Basu, [3]Naveen Raj Shukla

Abstract--*Image colorization is the mechanism of proceeding with a gray scale image as input and delivering colorized image as output that personifies the acceptable colors and tones of the image input (for instance, a sky on a clear sunny day must be obviously "blue" – it can't be colored "red" by the model). The approach we will use relies on deep learning. We will use a Convolutional Neural Network able enough of colorizing black and white images with results. We'll also apply object detection using which we will be able to know not only what is in the image but also where an object resides.*

Key words-- *Image colorization, black and white images.*

I. INTRODUCTION

In the following project we aim to build a machine learning model that will facilitate automatic colorization of a gray scale image into a colored one. Moreover, we will also detect the number of objects that are present in our image. Earlier approaches to colorization of black and white images depended on human manual annotation and very rarely produced results that were believable as true colorization. We have decided to counter the problem of image colorization using Convolutional Neural Networks to visualize what grayscale input would look like when colorized. Earlier Convolutional Neural Network bought about a great revolution in the field of computer vision. With each passing year the ImageNet Challenge has seen a sharp decline in its error rates due to the universal acceptance of CNN models amongst all the contestants. When it is about deep-learning based object detection there are majorly three primary object detectors you'll encounter namely R-CNN and their variants including the original R-CNN, fast R-CNN and faster R-CNN, SSDs (Single Shot Detector), YOLO. YOLO compared to other region proposal classification (fast R-CNN) which perform detection on various region proposals and thus end up performing prediction a whole lot of multiple times for the different regions in the image, it's architecture is more like fully convolutional neural network and passes the image (n x n) once through the FCNN and output is (m x m) prediction. The architecture splits the input image into m x m grid and for each grid generation two bounding boxes and class probabilities for those bounding boxes.

II. CHARACTERISTICS OF THE PROBLEM

We aim to ascertain that has 3 pixel values from a gray scale image having one value only. As mentioned earlier we will be having 3 pixel values i.e. instead of working with the RGB format we will be using the LAB (Lightness, A and B) colour space for our model. We prefer to use this over the RGB model because this makes it

[1]*Department of Computer Engineering SRM Institute of Technology, KTR, sreekumar.k@ktr.srmuniv.ac.in*
[2]*Department of Computer Engineering SRM Institute of Technology, KTR, souryabasu_supratik@srmuniv.edu.in*
[3]*Department of Computer Engineering SRM Institute of Technology, KTR, naveenraj_shukla@srmuniv.edu.in*

DOI: 10.37200/IJPR/V24I8/PR280116
Received: 21 Jan 2020 | Revised: 08 Feb 2020 | Accepted: 14 Mar 2020

easier for us to separate out the lightness channel from the other two(A and B). For predicting the values of the input pixels directly we will be making use of regression.

Talking about object detection YOLO predicts multiple bounding boxes per grid cell. At training time we only want one bounding box predictor to be responsible for each object.

We assign one predictor to be “responsible” for predicting an object based on which prediction has the current IOU with the ground truth. This leads to specialization between the bounding box predictors. Each predictor gets better at predicting certain sizes, aspect ratios or classes of object, improving overall recall.

III. HYPER COLUMN

This paper introduces us to the idea of ”hyper- columns” in a CNN. A hypercolumn for a pixel in the input image is nothing but a vector of all the activations above present that pixel. The way we implemented this was by forwarding an image through the VGG network and then removing a few layers, upscaling them to the original image size, and concatinating them all together. Hyper- columns normally uses the output of the last layer as feature representation. However the information in this layer is too coarse spatially to allow precise localization. Hypercolumn at a pixel is the vector of activation of all CNN units above that pixel. By this mean, spatial location information can be extracted from earlier layers and have a much more accurate prediction result. The hyper- column problem setting is as follows. First it is assumed we obtained a setoff detection system, af- ter non-maximum suppression (NMS). Then, the bounding box of detection is widened slightly and predicts a heat map on this expanded box. For segmentation, the heat map encodes he probability that a particular location is inside the object. Since that is segmented after detection, thus, we declare it as an instance segmentation approach at the title. And it also works for keypoint prediction, a separate heatmap is predicted for each keypoint, where each heatmap is the probability a location belongs to that keypoint. In each case, a 50*50 heatmap is predicted and then resized to the size of the expanded bounding box and splat onto the image.

IV. OUR MODEL

Similar to the RGB color space the lab color space has three channels, the L channel encodes lightness intensity, a channel encodes green-red, b channel encodes blue-yellow. Since the L channel encodes only the intensity, we can use the L channel as our grayscale input to the network. From there the net- work must learn to predict the a and b channels. Given the input L channel and the predicted ab channels we can then form our final output image. The whole mechanism can be outlined as trans- forming the training images from the RGB color space to Lab color space. Utilize the L channel as the input to the network and then train the net- work to predict the ab channels. Then combine the predicted ab channels with the input L channel. Then convert the Lab image back to RGB. Pixel classification is done using hypercolumn for object detection. A hypercolumn at a location is defined as one long vector concatenated the features from some or all of the feature maps in the network.

For instance, using pool2 (256 channels), conv4 (384 channels) and fc7 (4096 channels) from the architecture of AlexaNet would lead to a 4736 dimensional vector. Also location is necessary, for instance for a detected person his head should be on the top of the box. Hence the easiest way is to separate classifiers for each of

50*50 locations, that is using 1*1 convolution or fully connected (FC) layer on each hypercolumn can be used at each location. But there are three problems firstly, the amount of data passing through one point is few, which might lead to overfitting. Secondly, training such many classifiers are computational expensive. Thirdly, the adjacent pixels should be similar to each other. Now coming to efficient hypercolumn one solution is to use convolution and upsampling. The 1*1 convolution is replaced by n*n convolution. This corresponds to looking not only at unit directly above a pixel but also the neighborhood of the unit. Now in fast hypercolumn CNN are run just once on the whole image. The convolutional features can be shared among all boxes. Then for each box, the Spatial Pyramid Pooling (SPP) layer uses a spatial pyramid grid to calculate a fixed length vector that is then passed to the fully connected layers.

V. TOOLS USED

A. Spyder

It is a strong scientific environment for python, written in python, and designed for and by data analysts, engineers and scientists. It proposes a distinguish mixture of advanced profiling, debugging, analysis and editing functionality of a development tool.

B. NumPy

It is mostly used for applications involving concepts like tongue processing. PyTorch has been developed by Facebook's artificial-intelligence research group using Uber's ”Pyro” software for the concept of in-built probabilistic programming.

C. OpenCV

Adaptive Moment Estimation (Adam) keeps separate learning rates for every weight similarly as an exponentially decaying average of previous gradients. This combines elements of Momentum and Adagrad together and is fairly memory efficient since it doesn't keep a history of anything (just the rolling averages). it's reputed to figure well for both sparse matrices and noisy data. Adam seems promising for the exchange data.

D. YOLO Object Detector

YOLO is used here to detect objects. Here we use a single CNN to predict multiple bounding boxes and also calculate the class probability of those boxes. It trains on full images and directly optimizes the detection performance.

YOLO works in the following steps :- 1.resizing phase - input image to 448x448, 2. Then it works a single convolutional network on the image 3.Finally,it thresholds the resulting detections by the model's confidence.

VI. RESULTS

Firstly segmentation after detection, for hypercolumn 10*10 grid is used, also an additional 1/0 is summed according to whether location was inside region candidate or not. With variant grid sizes for discreting the detection box are tried: Using 1*1 grid already the SDS is outperformed. Full performance as the 10*10 grid is recovered using 5*5 grid. Keypoint is predicted where hypercolums obtains the results with fine tuning. Once the hypercolumn calculation is done, we proceed with the colorization model. Our neural network after predicting the colors, save a copy of the output on the local storage. This image now acts as an input for the YOLO object detection module. Once the objects are detected, the corresponding outputs are displayed.

VII. CONCLUSION AND FUTURE SCOPE

We made use of CNN to colour our black and white images of any dimension and add the object detection feature to the output. We have also extended the colorization process for videos and live input in the form of webcam. In the future, we will be ex- tending the object detection to the colorized video as this can make sure that the model would give a real time feed to color blind people.

VIII. ACKNOWLEDGEMENT

The authors gratefully acknowledge the management of SRM Institute of Technology, Kattankulathur for giving us the opportunity and a platform to implement our particular idea/project.

REFERENCES

1. Automatic Colorization Available: https://tinyclouds.org/colorize/
2. Project Report CS534 Available: http://pages.cs.wisc.edu/~marannan/ Project_Rep ort_CS534.pdf
3. Image Colorization with Deep Convolutional Neural Networks Available: http://cs231n.stanford.edu/reports/20 16/pdfs/219_Report.pd f
4. Colorizing B&W Photos with Neural Networks https//blog.floydhub.com/colorizing-b-w-photos-with-neural-network s/
5. Introduction to Loss Functions https:// blog. algorithmia. com/introduction-to-loss- functions/
6. Erhirhie, Earnest Oghenesuvwe, Chinwuba Paul, and . 2019. Edible Insects' bio-actives as anti-oxidants: Current status and perspectives.. Journal of Complementary Medicine Research, 10 (2), 89-102. doi:10.5455/jcmr.20190130100319
7. Morgado, M., Rolo, S., MacEdo, A., Pereira, L., Castelo-Branco, M.Predictors of uncontrolled hypertension and antihypertensive medication nonadherence(2010) Journal of Cardiovascular Disease Research, 1 (4), pp. 196-202. DOI: 10.4103/0975-3583.74263