Reinforcement Learning for Image Quality Improvement for Object Detection[*]

Siddharth Nayak

ee16b073@smail.iitm.ac.in

1 Introduction:

Object Detection performance for a deep network depends a lot on the image quality. Generally images are preprocessed by motion de-blurring, gaussian blurring to remove noise, etc. Not quite a lot of work has been done before on improving the images at the time of it getting captured. A photographer while taking images changes a lot of parameters like shutter speed, gains, aperture to get the best image. Bychkovsky et al [1] worked on training a network to come up with image retouches which look pleasing to a human by training it on a dataset which was digitally annotated by expert photographers using photoshop.Yang et al [2] worked on using reinforcement learning to get user based adaptive exposure control to come up with good shutter speeds to capture good images which looked pleasing to humans. This project aims towards making a camera capture images in real-time which are good for a given object detection neural network to increase its performance.

2 Motivation of the project:

Whenever an object detection network is trained, it is generally from a dataset which has images which have been taken with only one configuration of the camera parameters in cityscape conditions. But sometimes the cars may travel faster and there may be some motion blur in the captured image due to the motion of the camera as well as the motion of the other cars. Thus there may be some cases where the object detector is not confident about detecting an object due to motion blur. This is where a trained RL[1] agent will try to map the image captured into an image which is quite similar to the images in the dataset and makes the object detector confident about its detection.For example, if the dataset contains only images which have slow moving cars without blur in them and if while testing we encounter an image which has fast moving cars then the RL agent would make appropriate changes to the settings which would map it to a slow moving car type image.Also making bounding boxes is a bottleneck in obtaining training datasets. Thus when an RL agent is used the person annotating can just give rewards as +1 or 0 according the the quality of the image captured or we can have some heuristic to give rewards. And thus making the process faster.

3 Phase 1: Digital Image Transformation

Phase 1 was carried out from September 2018 to November 2018. In this phase we changed the brightness of images from the CIFAR-10 dataset. The reinforcement learning agent is supposed to transform the randomly changed(brightness) image to an image which is close to the original one.

3.1 Modelling of the MDP:

We use 64x64x3 images from the standard CIFAR-10 Dataset. The actions are changing the brightness of the image. While training the image is digitally altered by changing its brightness randomly. This randomly changed image is the state for the reinforcement learning agent. The amount of change in brightness is determined choosing one of the 40 values in $[0, 2]$ in steps of 0.5. Here choosing the action with value '1' does not change the image, choosing '0' gives a completely dark(black) image and choosing '2' gives a completely bright(white) image. Two different types of reward systems were tried out:

Human Reward System: A human gives these rewards by manually comparing the original image and the agent transformed agent

$$R = \begin{cases} +1 & \text{, if image almost similar to original} \\ 0 & \text{, otherwise} \end{cases}$$

Automated Reward System: the rewards is calculated using the following heuristics

$$R = \begin{cases} SI - \dfrac{(dr + dg + db)}{10000} & \text{, if similarity index} \geq 0.85 \\ -\dfrac{(dr + dg + db)}{10000} & \text{, otherwise} \end{cases}$$

Here SI is the similarity index between the original and the transformed image; dr,dg,db are the mean square errors in each of the RGB color channels and the factor of 10000 is just for normalising the reward.

[*]Rough Draft of the work.

[1]RL:Reinforcement Learning is used interchangebly

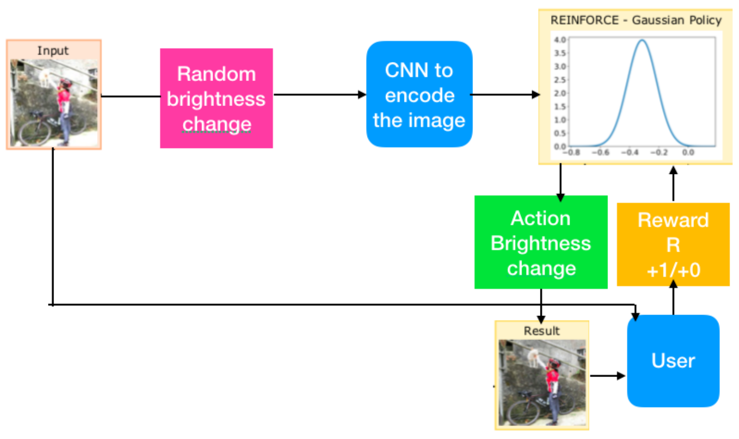


Figure 1: Block Diagram for phase 1 with human based rewards [2]

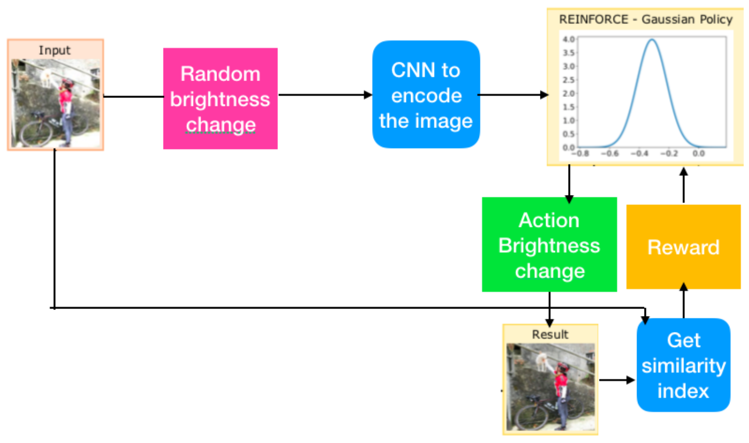


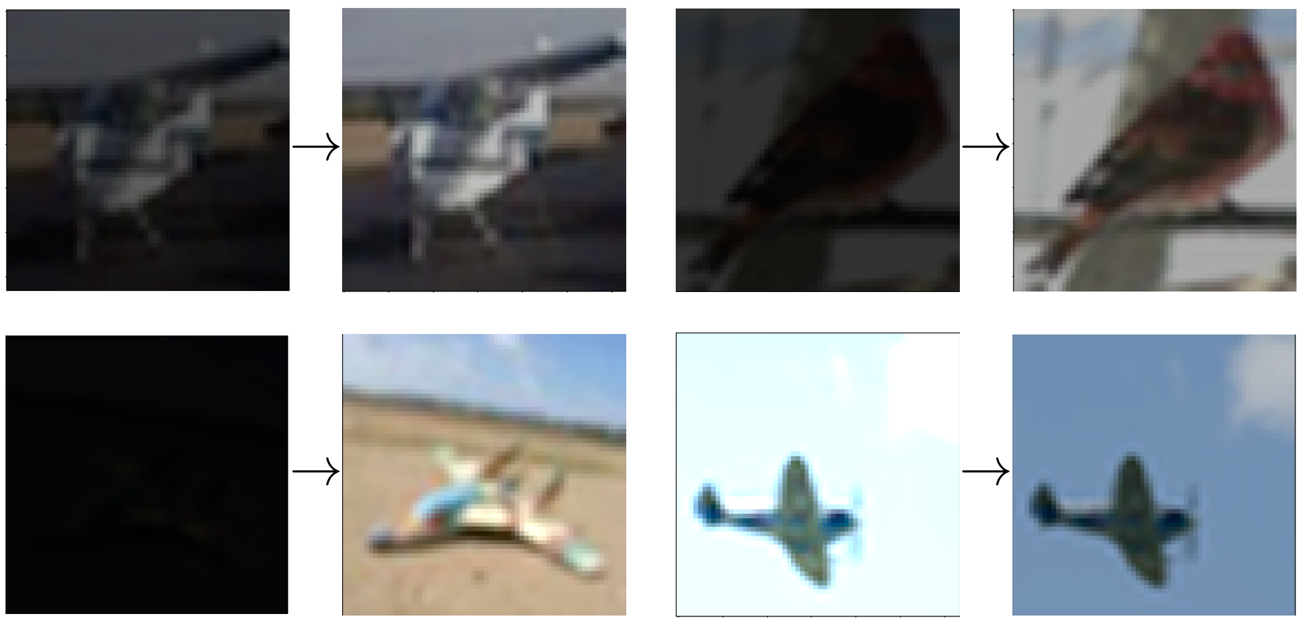
Figure 2: Block Diagram for phase 1 with automated rewards

3.2 The network

We are using the REINFORCE algorithm. The network is a 4 layered convolutional neural network with two fully connected layers.The actions include changing the brightness of an image by a certain factor.Here, if factor is zero then the image is completely black and if the factor is 2 then image is completely white. So the agent has to choose a value between 0 and 2 for changing the brightness of the image. Two different models were trained by giving in rewards manually as well as automated respectively. Though the manual reward system is quite ambiguous, depending on the person giving the rewards, we wanted to check the feasibility of the algorithm and its convergence properties. The network started taking good actions where it was not changing the image for the unaltered images in the dataset. Also it was taking the optimal actions for the dark as well as the bright images by making them bright and dark respectively. We further also tried working with the automated reward system and this approach made the model converge faster than the human based reward system.

4 Few of the sample actions the agent took

As seen below, the agent takes actions which convert dark images and bright images to ideal ones.



5 Phase 2: Adding the Object Detector

Phase 2 was carried out from December 2018. In this phase the object detection module was added. The pipeline of the training modules is as follows:

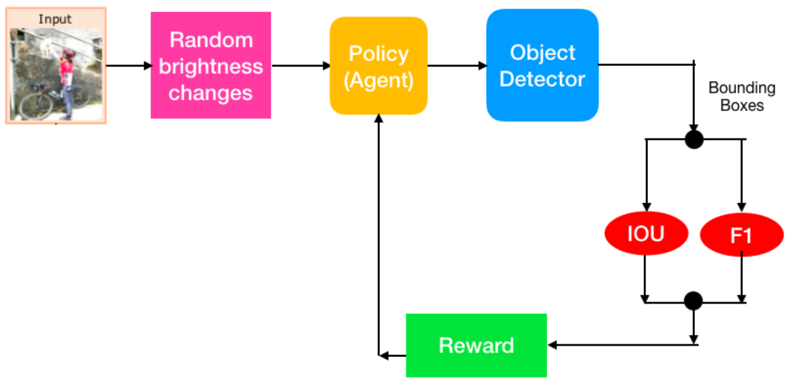


Figure 3: Block Diagram for phase 2 with the object detector

Algorithm 1 Training Procedure

```
1:  described here is one epoch of training :
2:      randomly shuffle the training set
3:      for image in dataset:
4:          state = random brightness change
5:          action = policy(state)
6:          new_image = change brightness of state according to action
7:          bounding_boxes = detector(new_image)
8:          IOU, F1 = eval_bounding_boxes(bounding_boxes)
9:          reward = $\alpha.(IOU) + (1-\alpha).(F1)$
10:         backpropagation
```

So in an epoch we randomly shuffle the dataset first. The image is first digitally modified randomly by changing the brightness. Now this modified image is the state for the reinforcement learning agent. Now this state is given as the input to the policy network which is a Resnet-18. The policy network gives out a probability distribution over the possible actions and the action to be taken is sampled out of this distribution. This action is applied onto the state to get the agent-modified image. This image is the input for the object detector. As of now we have tried out the model with YOLO as the object detector. The reward is the weighted sum of the Intersection over Union(IOU) and the F1 score of the bounding box predictions of the image. So the reward is as follows:

$$R = \alpha.(IOU) + (1-\alpha).(F1)$$

Here we set $\alpha = 0.5$ for the initial experiments. We call the baseline model as the performance of the object detector without random digital changes in the image. The baseline has an average reward over the dataset of 0.401 and the agent has reached a average reward of 0.395.

6 Phase 3: Future work and possible improvements

This phase will be carried out under the Undergraduate Research Course at the Indian Institute of Technology Madras during the Spring 2019 semester for credit.
In this phase we would like to add more action possibilities like changing the contrast, brightness, sharpness, etc. Also we realised that YOLO being a poor object detector (in terms of accuracy) we would like to try out better models like FRCNN as the object detector. Also another thing which could be tried out is changing the reinforcement learning algortihm from REINFORCE to other policy gradient methods for faster convergence. Once the algorithm proven to converge on post processing of the images, we would also like to work on getting optimal shutter speeds and voltage gains using a similar approach.

7 Acknowledgements

I would like to thank Manan Tomar and Rahul Ramesh, two of my seniors and Prof. Balaraman Ravindran(Guide) who helped me out in my project with their valuable suggestions. I would also like to thank RISE Lab, Indian Institute of Technology Madras for providing the computing resources.

8 References

[1] Learning Photographic Global Tonal Adjustment with a Database of Input/Output Image Pairs
[2] Personalized Exposure Control Using Adaptive Metering and Reinforcement Learning