International Journal of Innovative Technology and Exploring Engineering (IJITEE)
ISSN: 2278-3075 (Online), Volume-9 Issue-6, April 2020

Optimization of Feature Loss for Image Enhancement

Vishal Burman, Harshinee Sriram, S. Usha Kiruthika

Abstract: ***Image-transformation problem is a problem in which an input image is transformed to an output image. In most of the recent methods, a feed-forward neural network is defined which utilizes per-pixel loss between the output image and the ground-truth image. In this paper we have showcased that high-quality images can be generated by defining a feature-loss function which is based on high-level perceptual features extracted from pre-trained convolutional networks. We have combined both the approaches that have been formerly mentioned and have proposed a feature-loss function for training a feed-forward neural network capable of image transformation tasks. We have compared out method with that of an optimization based approach, similar to the one utilized in Generative Adversarial Networks (GANs) and our method produced visually appealing results whilst fully capturing the intricate details of the object in the image.***

Keywords: ***image transformation, super-resolution, deep-learning, convolutional neural networks***

I. INTRODUCTION

A classical problem can be well defined as an image transformation problem if the system is provided with an image and the output that is computed by the system is a transformed variation of the initial image. Some well-known examples are de-noising, image-enhancement etc. These examples contain an input, which is essentially a degraded image, and the corresponding output is a high-quality image. Some known examples in the domain of Computer Vision are semantic segmentation and depth estimation, each of which contains an input image and the corresponding output image encodes the per-pixel semantic and per-pixel geometric data about the scene.

One particularly common method to solve an image transformation task is to train a feed-forward convolutional neural network which allows a per-pixel loss method to measure the difference between the output image and the corresponding ground-truth image. This method was incorporated by Dong [11] for image-enhancement. This method has been observed to be efficient at test-time because it only requires a forward pass through the convolutional neural network.

Revised Manuscript Received on April 30, 2020.
***** Correspondence Author
Vishal Burman*, Bachelor of Technology, Computer Science and Engineering at SRM Institute of Science and Technology, Kattankulathur
Harshinee Sriram, Bachelor of Technology, Computer Science and Engineering at SRM Institute of Science and Technology, Kattankulathur
S. Usha Kiruthika, Assistant Professor, Department of Computer Science and Engineering in SRM Institute of Science and Technology, Kattankulathur

However, the method prescribed above does not include the feature differences between the output image and the ground-truth image.

An example for the same can be as follows, consider two images which are seemingly identical and only differ from each other with respect to a single pixel.

These two images, hence, are similar feature-wise but are very different if they were to be compared by the per-pixel loss method.

Recent work has shown that high-quality images can be obtained by using feature-loss method, and this method is not based on per-pixel difference, but instead it relies on the difference between the high-level features that are extracted from the pre-trained convolutional neural network.

In this method, the images are obtained by minimizing a loss function. This method has been previously incorporated in inversion of features to find intermediate features captured by several layers of convolutional neural networks and also in style transfer by Gatys [12].

The above methods produce very high-quality images but are often quite slow, owing to their solving of the optimization predicament.

In this paper, we combine the advantages of both the methods have been formerly mentioned. In our method, we have trained a feed-forward neural network for the image-transformation task, but instead of relying on per-pixel loss, which heavily relies on only the difference between the pixels of the output and input/ground-truth images, we have trained our feed-forward transformation neural network utilizing feature-loss functions.

We observed that feature-loss calculates image similarities more accurately than the previous per-pixel loss method. We have conducted extensive experiments on image-enhancement.

The predicament of image-enhancement is very ill-posed. In practice, for image-enhancement, there can be a plethora of high-resolution images that can generate the same degenerated image.

However, by all-encompassing experimentations, we have discovered that we need not learn features from scratch because by the utilization of feature-loss function we can transfer the required semantic knowledge from the loss network to the feed-forward transformation neural network.

II. LITERATURE SURVEY

In our literature survey, we looked at research that was done on super-

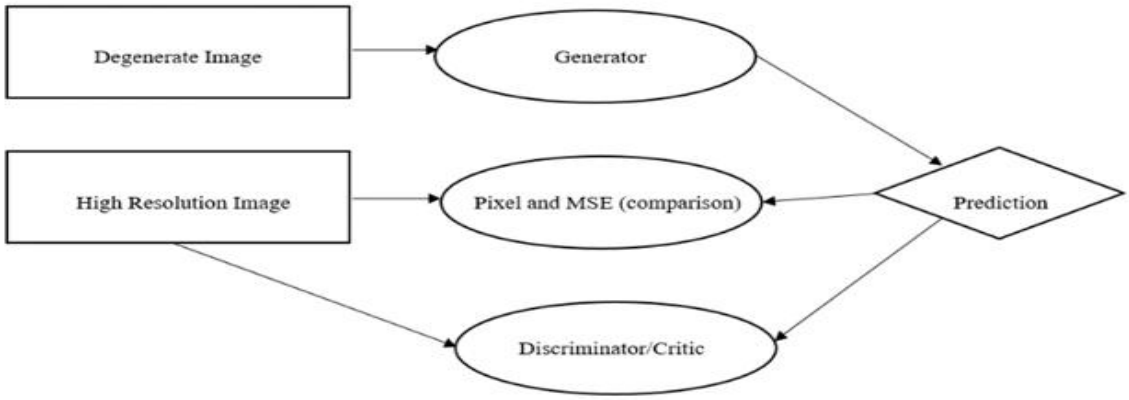

Fig. 1. Architecture of the workflow prescribed by ESRGAN (Enhanced Super-Resolution Generative Adversarial Networks)

resolution, depth map prediction, deep image representations, scene labeling, depth estimation, semantic segmentation, and deep colorization as these subject matters dealt with either retrieving lost information or providing more clarity to a pre-existing range of information. However, whilst studying these papers it was evident that, with respect to image enhancement, there existed certain predicaments which were inherent to the architectures or algorithms provided by them.

A major disadvantage of using a method underpinned by Generative Adversarial Networks (GANs) is the inability of per-pixel losses to capture intricate or localized feature differences that exist between the generated or output images and the ground-truth, or simply, ground images. In addition, with respect to the Enhanced Super-resolution Generative Adversarial Networks (ESRGAN) [1], the use of relativistic GAN results in a training time that is prolonged. Delving into [2], which utilizes two deep network stacks for super-resolution, it is observed that depth maps not being provided in the resolution of the original input image is a paramount setback. The method proposed in [3] fails due to the lack of stability that is preferred when altering hyper-parameters of the network.

Using Deep Convolutional Neural fields, as proposed by [5], uses support vector regressor (SVR) which is not optimized for images that are high quality in nature. The method provided by [8] to retrieve lost information is also ruled out because it requires training on an enormous dataset which must ideally encompass all possible objects and the method is invalid for synthetic images. Utilizing fully convolutional networks, as described by [11], for the purpose of semantic segmentation possesses the disadvantage of feature-loss with every convolutional operation – which in turn leads to a potential loss of features and hinders the path to the intended goal.

III. METHODOLOGY

In order to understand the difference of approach, the methodology prescribed by the base paper, [1], must be inspected. The implementation in [1] begins with obtaining the IIIT Pet Dataset which contains high resolution images. Then, degenerate images from these high resolution images are devised. The values of hyper-parameters are: batch size of 32 and image size of 128x128. Next, the generator is pre-trained. In this segment, the learner architecture utilized is the U-Net model and the architecture of Resnet-34. Normalization of images is done by using ImageNet stats, and the loss is defined by mean squared error.

The next stage is the training loop which has two phases, the initial phase where the model is fit for the premier two cycles, and the second phase where layers are unfrozen and trained for three cycles. Then, the images that have been generated are saved. Post this step, the critic is trained with the following hyper-parameters: batch size of 32, image size of 128x128, loss deduction by primary cross-entropy. The images used for this stage are those created by the generator. The training loop trains this newly formed critic for six cycles.

Next, a GAN is formulated using the generator and the critic, containing the Adam optimizer and a learning rate of 10^-4. The training loop trains this GAN for forty cycles. After this, the image size is increased and its new dimensions are 192x192. These are then utilized to train the GAN for ten cycles. Fig. 1 showcases the general workflow of this approach. Our workflow, in turn, is depicted in Fig. 2. The method we propose comprises of two parts which are: image transformation network and a loss network (which consists of several individual loss functions l1, l2, l3… lk). The image transformation network is a deep convolutional neural network, which encompasses weights W.

The network transforms input images x into output images y' through y' = f(x). Each loss function then calculates the differences between the output image and the input/ground-truth image. The image transformation network exhibited in the base paper is trained utilizing stochastic gradient descent to minimize the weighted combination of the loss function. This has been portrayed by the equation (1).

$$W^* = \arg min_W E_{x,\{y_i\}}[\sum_{i=1} \lambda_i l_i(f_W(x), y_i)] \quad (1)$$

The key insight arrived from recent works in computer-vision and deep convolutional neural networks, in which pre-trained networks inherently comprise of encoded features and semantic information required by us in our loss function.

For the purpose of image enhancement, input x is a degenerated image and the target is a high-resolution image.

A. Image Transformation Network

We avoid the incorporation of pooling layers, and instead, we use strided and fractionally strided convolutions for our in-network down-sampling and un-sampling. Our network is built with five residual blocks (the structure of these blocks are defined by ResNets), and followed by a residual block with a layer of batch normalization.

We use ReLU (Rectified Linear Unit) activation functions with the exception of the output layer, which utilizes the tan h function for activation. The premier and final layers use 9x9 kernels whereas all other convolutional layers use 3x3 layers.

B. Input and Output

For image enhancement, with an up-sampling factor f, the output is a high-resolution patch of shape 3X288X288 and the corresponding input is a degenerated image of shape 3X288/fx288/f.

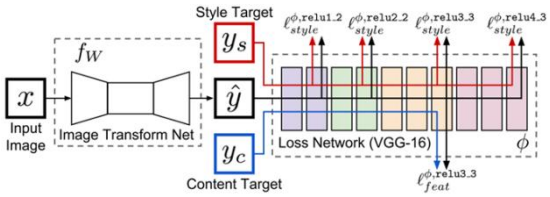

Fig. 2. Proposed workflow showcasing the stages and imperative variables

C. Down-sampling and Up-sampling

With an up-sampling factor f, several residual blocks are utilized followed by log2 (f) convolutional layers, each with a stride of ½. This allows the up-sampling function to be learnt in concurrence with the remaining part of the network.

D. Feature Loss Function

Each of our feature loss functions are deep convolutional neural networks by themselves. In our experiment, the loss network is the 16 layer VGG network that has been pre-trained on ImageNet.

E. Feature Reconstruction Loss

Rather than trying to make the pixels of the output image to be exactly equal to those of the input/ground-truth image, we encourage them to have similar feature representations. The feature reconstruction loss is the Euclidean distance between the feature representations. This has been showcased in (2).

$$l^{\phi,j}_{feat}(\hat{y}, y) = \frac{1}{C_j H_j W_j} \left|\left|\phi_j(\hat{y}) - \phi_j(y)\right|\right|^2_2 \qquad (2)$$

As we reconstruct higher layers, the core image content and overall spatial structure is maintained but the colour, texture, and the shape of the image is not maintained. Therefore, using feature reconstruction loss, it encourages the image to be similar when it comes to feature-wise comparisons but it does not force them to be a perfect match.

F. The Experimentation Process

In image-enhancement, the main objective is to generate a high-resolution enhanced image from a previously degenerated image or a low-resolution image. This is highly ambiguous and ill-posed problem because for a single degenerated or low resolution image there can be multiple high-resolution enhanced images which would have the capability to generate it. This ambiguity of the problem becomes much more apparent as the enhancement factor grows; for example in large enhancement factors(x4, x8), the intricate details of the enhanced image may have little to no evidence in its corresponding low-resolution version of the image.

To overcome this ambiguous and ill-posed problem, we have trained our network omitting the per-pixel loss method which is commonly used for this type of problems and instead replaced it with feature-reconstruction loss which allows us to transfer the semantic knowledge of the input image from the pretrained VGG network to that of our image-enhancement network. We have exclusively focused on larger enhancement factors(x4, x8) since larger factors requires more semantic reasoning from in the input image.

The methods which are commonly used while comparing models for image-enhancement are SSIM and PSNR. These models often prove poor metrics and poorly correlate with human equivalent assessment of visual quality of the images. The above two mentioned methods largely use the per-pixel low-level difference method to assess the quality of the image moreover the PSNR operates under the assumption that the image has been added with Gaussian noise. In addition to this PSNR relies on the method of per-pixel loss so models which are trained on minimizing the per-pixel loss will always outperform the models which are trained to minimize the feature-reconstruction loss. Therefore the emphasize of this experiment is not to achieve state-of-the-art result on PSNR and SSIM but to show the perceptual and quality difference between models trained using per-pixel loss method and feature reconstruction loss method.

We have trained the model to perform on enhancement factors such x4 and x8. This is achieved by minimizing the feature reconstruction loss method incorporated in our experiment. The feature-reconstruction loss method is applied at relu_2 of the VGG-16 loss network ϕ(phi). We have trained the with 288x288 patches with images obtained from IITD pets dataset, and prepare the low resolution input images by blurring out the high definition images using a custom made function which utilises a Gaussian kernel of width 0.1. We apply downsampling to the images by using bicubic interpolation.

1cycle policy=> In order to further improve upon the training time, we have utilised the 1cycle policy which was introduced by Leslie Smith. Instead of using a fixed or decreasing learning rate, we oscillate between between reasonable minimum and maximum bounds.

1cycle consists of two steps; one in which the learning rate increases and one in which the learning rate decreases. Each step has a size(called stepsize) which is the number of iterations where learning rate increases or decreases. Two steps form a cycle.

1cycle policy are not computationally expensive and eliminate the need to find the best learning rate value-the optimal learning rate will fall somewhere between the minimum and maximum bounds.

Compared to other methods our model which is trained using feature reconstruction loss does a fine job at improving sharp edges and fine details of the image. Many of the results from our models have grid-like artefacts at a pixel level which hurts the PSNR and SSIM scores. These artefacts are a result of the feature reconstruction loss method which is used in our models and is not a result of our transformation network. The pixel loss method gives fewer visual artefacts and higher PSNR values, but our model utilising feature reconstruction loss does a better job at reconstructing fine details which leads to visually pleasing results.

G. Technical walkthrough of architecture formation

The primary library utilized was fastai, and the major library modules incorporated were vision and callbacks. The VGG16 model was also imported for high level feature generation.

The preliminary tasks included assigning paths to the train, validation, and test image classes and to carry out pre-network image transformations on the stream of incoming images - which comprised of creating the appropriate directories, defining the dimensions of the target image, and saving the resulting image.

Post this step, a set was created where it was checked if all necessary images have been included in the appropriate directories, and if that were not the case, the outliers were resized and reshaped and then assigned to the most appropriate directory.

Next, the values of batch size and image size were initialized to 16 and 128 respectively. The resnet34 model was stored in a variable termed as "arch". Next, the stream of reformed input images were split by random and stored in the image lists - which were about to be fed into the personalized architecture.

At this point, a module named "get_data" was defined, which took an image, applied a zoom factor, altered the size to the previously assigned value of 128, transformed it and normalized the image.

This resulting image was then to be returned by the module. The next module defined was "gram_matrix" which is utilized to compute the Hermitian matrix pertaining to inner products of a set of vectors that reside in an inner product space.

The base loss is initialized post this step. This brings attention to the next phase that revolves around the pre-trained network.

The VGG16 network is attuned to personalized specifications, and then a variable termed as "blocks" is initialized with the children segments of the VGG network that showcase the MaxPool characteristic.

This, in all cases, resulted in layers with rectified linear unit activation with the “inplace” value set to "true". Up next, a class was created under the name "FeatureLoss". The constructor of the class took into account the feat, loss features, hooks, weights, as well as metric names.

A second class function was invoked to make features and the third class function named "forward" defined the necessary updates that would be applied to the feature losses and metrics as the processing of the images progressed.

In the last function self-hooks were removed in the definition.

A variable named "feat_loss" was created to contain the attuned VGG network passed through the FeatureLoss class. The "unet_learner" module was passed to a variable named "learn" and this contained the data, architecture, loss function, callback functions, blur, and normalization type as parameters.

The learning rate hence deciphered was visualized through a plot with a fairly negative slope progression. Fig. 3 shows this progression.

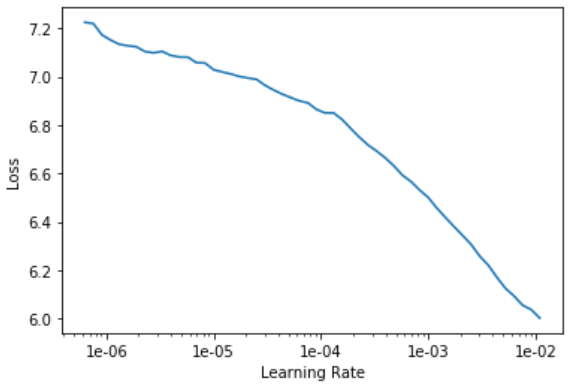

Fig. 3. The Progression of Learning Rate

Up next, a function called "do_fit" was created, which fit the images in accordance to the most effective learning rate, and callbacks were initiated to monitor the progress. The monitored progress is portrayed in fig. 4 and fig 5. The learn segment was unfreezed and then the sliced layers were fit accordingly. The next stage was testing. For this, all modules of the utility section in

epoch	train_loss	valid_loss	pixel	feat_0	feat_1	feat_2	gram_0	gram_1	gram_2	time
0	2.140367	2.116270	0.166983	0.258380	0.286314	0.149623	0.346866	0.552237	0.355867	04:52
1	2.144754	2.115357	0.167109	0.258478	0.286276	0.149554	0.346475	0.551651	0.355815	04:53
2	2.132632	2.114127	0.167676	0.258793	0.286237	0.149507	0.345054	0.551281	0.355579	04:53
3	2.111077	2.110781	0.167101	0.258186	0.285610	0.149262	0.345397	0.550147	0.355078	04:53
4	2.118616	2.110332	0.167279	0.257998	0.285398	0.148879	0.345353	0.550711	0.354714	04:53
5	2.116986	2.107983	0.167103	0.258148	0.285357	0.148693	0.344830	0.549800	0.354051	04:53
6	2.118927	2.107269	0.167103	0.257970	0.285133	0.148486	0.344966	0.549753	0.353857	04:53
7	2.130412	2.106136	0.167416	0.258193	0.285224	0.148629	0.343758	0.549128	0.353788	04:54
8	2.124623	2.105314	0.167200	0.257998	0.285187	0.148677	0.343914	0.548584	0.353754	04:53
9	2.123753	2.105761	0.167286	0.258133	0.285274	0.148584	0.343758	0.549081	0.353645	04:53

Fig. 4. Monitored values in the premier set of training

epoch	train_loss	valid_loss	pixel	feat_0	feat_1	feat_2	gram_0	gram_1	gram_2	time
0	3.871685	3.645456	0.137035	0.224914	0.309442	0.219027	0.548641	1.184249	1.022148	08:24
1	3.768753	3.592071	0.149521	0.224917	0.307207	0.216489	0.516586	1.166191	1.011161	08:17
2	3.746760	3.540148	0.140845	0.225902	0.305615	0.212939	0.506178	1.153607	0.995063	08:17
3	3.696663	3.428905	0.140778	0.223331	0.297759	0.205318	0.490568	1.111482	0.959670	08:18
4	3.598644	3.391242	0.136184	0.221870	0.293767	0.201873	0.491534	1.099982	0.946032	08:19
5	3.516614	3.377638	0.140705	0.220042	0.291552	0.200217	0.488529	1.098604	0.937989	08:18
6	3.522672	3.335736	0.142141	0.220847	0.291331	0.199064	0.467310	1.077259	0.937784	08:18
7	3.496808	3.327619	0.135593	0.219046	0.288970	0.197447	0.480221	1.079776	0.926565	08:18
8	3.510936	3.307173	0.140445	0.220030	0.287702	0.195840	0.472268	1.077155	0.913733	08:17
9	3.389608	3.168043	0.136081	0.215299	0.279291	0.188602	0.436402	1.029009	0.883361	08:17

Fig. 5. Monitored values in the final set of training

fastai were imported the GPU memory was freed by clearing all existing cache.

The maximum size of the test image was made with respect to the available GPU RAM. The unet_learner was assigned to the "learn" variable and it contained the data, architecture, loss function, blue boolean, and normalization type in its parameters. The "data_mr" iterator was created that applied the necessary pre-network image alterations and transformations. After this, the images were tested by displaying them with a figure size of 18x15 and interpolation set to "nearest". The comparisons and deciphered results have been analyzed in the future sections.

IV. PERFORMANCE PARAMETRICS OF THE PROPOSED ARCHITECTURE

Fig. 3 and Fig. 4 show the comparison between the existing GAN workflow and our proposed workflow. In a feature-loss method, both the training loss and the validation loss decrease as the number of cycles of training increase. However, as can be observed, the training and validation losses converge and cease to decrease after a certain amount of batches in the GAN workflow, whereas in our case they continue decreasing. This also exhibits the fact that our model continues to better itself, given a constant number of processing batches, as opposed to converging and remaining stagnant, which can be observed to be a significant improvement when compared to the other model that exhibits the usage of GANs.

V. OUTPUT AND INFERENCES

Some instances of the output generated by our method are presented in Fig. 6 and Fig. 7. As can be observed, our method helps identify localized or domain-specific information (such as the eyes of the subject), which was a disadvantage that existed in the GAN approach.

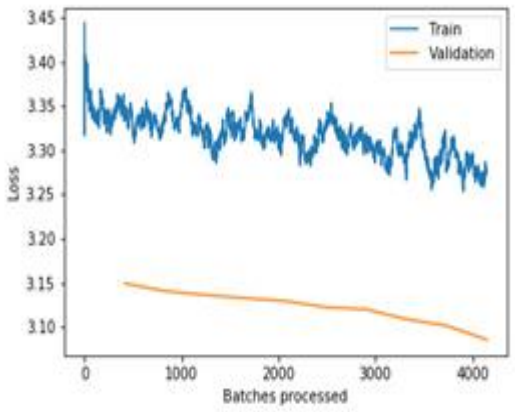

Fig. 6. Comparison between training and validation loss using the GAN approach

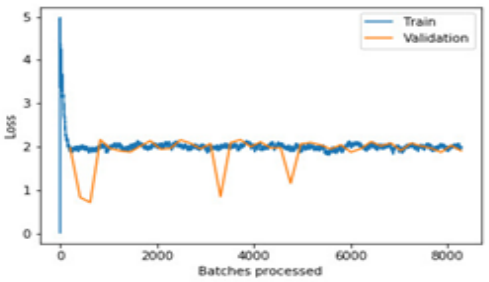

Fig. 7. Comparison between training and validation loss using the our approach

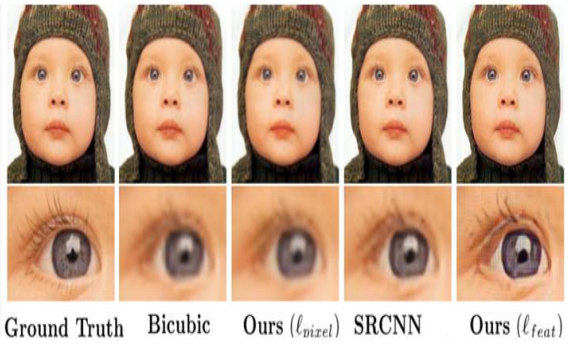

Fig. 8. Reconstructed image - output example

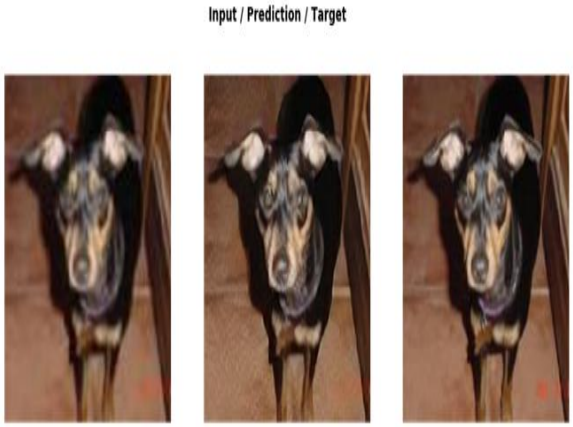

Fig. 9. Progression of image processing from ground-truth to generated output (with model comparisons)

We also overcame a challenge that dwelled in this network, which was to write custom callbacks to call the intermediate activations of the VGG network. Hence, compared to other methods, our model trained for feature reconstruction does a better job at reconstructing sharp edges as well as fine details.

However, one limitation that exists in our body of work is the extensive utilization of a large amount of GPU, which was required for experimentation on batch size and image size.

Thus, future improvements or upgrades could work on making this process more GPU-efficient, and optimizing the network so as to make it a comparatively lighter task.

Improving this aspect would potentially make the architecture more accessible and more mobile in nature for a plethora of super-resolution and image enhancement domains.

VI. CONCLUSION

This study establishes the fact that combining the two distinct approaches of image enhancement, namely creating a feed-forward neural network that uses per-pixel loss and a generative adversarial network (GAN) technique, works best when considering quality and accuracy, with one disadvantage being that the extensive resource utilization pertaining to GPU can cause a technological burnout.

In conclusion, using a loss network pre-trained for image classification to define feature loss functions that help in measuring perceptual differences in content and style between images is a more improved and effective method for achieving super-resolution and clarity, with the added advantage of detecting localized features that would have, otherwise, been looked over.

REFERENCES

1. Wang, Xintao, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Yu Qiao, and Chen Change Loy. "Esrgan: Enhanced super-resolution generative adversarial networks." In Proceedings of the European Conference on Computer Vision (ECCV), pp. 0-0. 2018.
2. Eigen, David, Christian Puhrsch, and Rob Fergus. "Depth map prediction from a single image using a multi-scale deep network." In Advances in neural information processing systems, pp. 2366-2374. 2014.
3. Mahendran, Aravindh, and Andrea Vedaldi. "Understanding deep image representations by inverting them." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 5188-5196. 2015.
4. Farabet, Clement, Camille Couprie, Laurent Najman, and Yann LeCun. "Learning hierarchical features for scene labeling." IEEE transactions on pattern analysis and machine intelligence 35, no. 8 (2012): 1915-1929.
5. Liu, Fayao, Chunhua Shen, and Guosheng Lin. "Deep convolutional neural fields for depth estimation from a single image." In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5162-5170. 2015.
6. Vondrick, Carl, Aditya Khosla, Tomasz Malisiewicz, and Antonio Torralba. "Hoggles: Visualizing object detection features." In Proceedings of the IEEE International Conference on Computer Vision, pp. 1-8. 2013.
7. Dong, Chao, Chen Change Loy, Kaiming He, and Xiaoou Tang. "Image super-resolution using deep convolutional networks." IEEE transactions on pattern analysis and machine intelligence 38, no. 2 (2015): 295-307.
8. Cheng, Zezhou, Qingxiong Yang, and Bin Sheng. "Deep colorization." In Proceedings of the IEEE International Conference on Computer Vision, pp. 415-423. 2015.
9. Zhang, Richard, Phillip Isola, and Alexei A. Efros. "Colorful image colorization." In European conference on computer vision, pp. 649-666. Springer, Cham, 2016.
10. Long, Jonathan, Evan Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 3431-3440. 2015.
11. Dong, Chao, Chen Change Loy, Kaiming He, and Xiaoou Tang. "Learning a deep convolutional network for image super-resolution." In European conference on computer vision, pp. 184-199. Springer, Cham, 2014.
12. Gatys, Leon A., Alexander S. Ecker, and Matthias Bethge. "Image style transfer using convolutional neural networks." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2414-2423. 2016.

AUTHORS PROFILE

Vishal Burman is currently pursuing Bachelor of Technology in Computer Science and Engineering at SRM Institute of Science and Technology, Kattankulathur Campus. His research interest is Machine Learning and he is currently exploring machine learning and deep learning frameworks.

Harshinee Sriram is currently pursuing Bachelor of Technology in Computer Science and Engineering at SRM Institute of Science and Technology, Kattankulathur Campus, She is an undergraduate Artificial Intelligence researcher at Association of Computing Machinery's Special Interest Group for Artificial Intelligence and is a Visiting Research Scholar at the University of Auckland, New Zealand. She is also the author of the book titled "The Horrors of Happiness" and co-author of the book titled "Pandora's Box – Horn OK Please". She has delivered technical talks at various conferences all over the world. Her current research interests revolve around finding artificially intelligent solutions for mental health and well-being.

S. Usha Kiruthika is currently working as an Assistant Professor in the Department of Computer Science and Engineering in SRM Institute of Science and Technology, Kattankulathur Campus. She has completed her Ph.D. at Anna University and she has many publications to her credit. Her areas of interest include artificial intelligence and deep learning.